**DATA ANALYTICS LAB**

|  |  |
| --- | --- |
| EX:NO:1 | **VECTORS,LISTS,ARRAY,MATRIX,DATA FRAMES** |
| DATE: |

**1.VECTORS**

# Create a vector.

apple <- c('red','green',"yellow")

print(apple)

# Get the class of the vector.

print(class(apple))

**OUTPUT**

> # Create a vector.

> apple <- c('red','green',"yellow")

> print(apple)

[1] "red" "green" "yellow"

>

> # Get the class of the vector.

> print(class(apple))

[1] "character"

**2.LISTS**

# Create a list.

list1 <- list(c(2,5,3),21.3,sin)

# Print the list.

print(list1)

**OUTPUT**

> # Create a list.

> list1 <- list(c(2,5,3),21.3,sin)

>

> # Print the list.

> print(list1)

[[1]]

[1] 2 5 3

[[2]]

[1] 21.3

[[3]]

function (x) .Primitive("sin")

**3.ARRAY**

# Create two vectors of different lengths.

vector1 <- c(5,9,3)

vector2 <- c(10,11,12,13,14,15)

# Take these vectors as input to the array.

result <- array(c(vector1,vector2),dim = c(3,3,2))

print(result)

**OUTPUT**

> # Create two vectors of different lengths.

> vector1 <- c(5,9,3)

> vector2 <- c(10,11,12,13,14,15)

>

> # Take these vectors as input to the array.

> result <- array(c(vector1,vector2),dim = c(3,3,2))

> print(result)

, , 1

[,1] [,2] [,3]

[1,] 5 10 13

[2,] 9 11 14

[3,] 3 12 15

, , 2

[,1] [,2] [,3]

[1,] 5 10 13

[2,] 9 11 14

[3,] 3 12 15

**4.MATRIX**

# Create a matrix.

M = matrix( c('a','a','b','c','b','a'), nrow = 2, ncol = 3, byrow = TRUE)

print(M)

**OUTPUT**

> # Create a matrix.

> M = matrix( c('a','a','b','c','b','a'), nrow = 2, ncol = 3, byrow = TRUE)

> print(M)

[,1] [,2] [,3]

[1,] "a" "a" "b"

[2,] "c" "b" "a"

**5.DATA FRAMES**

# Create the data frame.

emp.data <- data.frame(

emp\_id = c (1:5),

emp\_name = c("Rick","Dan","Michelle","Ryan","Gary"),

salary = c(623.3,515.2,611.0,729.0,843.25),

start\_date = as.Date(c("2012-01-01", "2013-09-23", "2014-11-15", "2014-05-11",

"2015-03-27")),

stringsAsFactors = FALSE

)

# Print the data frame.

print(emp.data)

**OUTPUT**

# Create the data frame.

> emp.data <- data.frame(

+ emp\_id = c (1:5),

+ emp\_name = c("Rick","Dan","Michelle","Ryan","Gary"),

+ salary = c(623.3,515.2,611.0,729.0,843.25),

+

+ start\_date = as.Date(c("2012-01-01", "2013-09-23", "2014-11-15", "2014-05-11",

+ "2015-03-27")),

+ stringsAsFactors = FALSE

+ )

> # Print the data frame.

> print(emp.data)

emp\_id emp\_name salary start\_date

1 1 Rick 623.30 2012-01-01

2 2 Dan 515.20 2013-09-23

3 3 Michelle 611.00 2014-11-15

4 4 Ryan 729.00 2014-05-11

5 5 Gary 843.25 2015-03-27

|  |  |
| --- | --- |
| **EX.NO:4** | **NUMERICAL AND CHARACTER FUNCTION,STATISTICAL FUNCTION** |
|  |

**1.NUMERICAL FUNCTIONS**

**(1)abs(x):**

**SOURCE** **CODE**

abs(5)

**OUTPUT**

> abs(5)

[1] 5

**(2)ceiling(x):**

**SOURCE** **CODE**

ceiling(12.3)

**OUTPUT**

> ceiling(12.7)

[1] 13

**(3)floor(x):**

**SOURCE** **CODE**

floor(12.5)

**OUTPUT**

> floor(12.5)

[1] 12

**(4)sqrt(x):**

**SOURCE** **CODE**

sqrt(36)

**OUTPUT**

> sqrt(36)

[1] 6

**(5)log(x):**

**SOURCE** **CODE**

log(123)

**OUTPUT**

> log(123)

[1] 4.812184

**(6)trunc(x)**

**SOURCE** **CODE**

trunc(5.99)

**OUTPUT**

> trunc(5.99)

[1] 5

**(7)round(x,digits=x)**

**SOURCE** **CODE**

round(5.467,digits=2)

**OUTPUT**

> round(5.467,digits=2)

[1] 5.47

**(8)log10(x):**

**SOURCE** **CODE**

log10(123)

**OUTPUT**

> log10(123)

[1] 2.089905

**(9)signif(x,digits=n):**

**SOURCE** **CODE**

signif(12.2345,digits=2)

**OUTPUT**

> signif(12.2345,digits=2)

[1] 12

**(10)exp(x):**

**SOURCE** **CODE**

exp(10)

**OUTPUT**

> exp(10)

[1] 22026.47

**(11)cos(x):**

**SOURCE** **CODE**

cos(0)

**OUTPUT**

> cos(0)

[1] 1

**(12)sin(x):**

**SOURCE** **CODE**

sin(0)

**OUTPUT**

> sin(0)

[1] 0

**(13)tan(x):**

**SOURCE** **CODE**

tan(0)

**OUTPUT**

> tan(0)

[1] 0

**URL:** <https://data-flair.training/blogs/r-numeric-and-character-functions/>

**2.CHARACTER FUNCTIONS**

**(1)substr(string,start value,stop value):**

**SOURCE CODE**

string="priya"

substr(string,2,4)

**OUTPUT**

> string="priya"

> substr(string,2,4)

[1] "riy"

**(2)strsplit(string,””):**

**SOURCE CODE**

strsplit("abc","")

**OUTPUT**

> strsplit("abc","")

[[1]]

[1] "a" "b" "c"

**(3) sub(pattern, replacement, x, ignore.case =FALSE, fixed=FALSE)**

**SOURCE CODE**

sub("\\s",".","Hello There")

**OUTPUT**

> sub("\\s",".","Hello There")

[1] "Hello.There"

**(4)toupper(string):**

**SOURCE CODE**

toupper("priya")

**OUTPUT**

> toupper("priya")

**[1] "PRIYA"**

**(5)tolower(string):**

**SOURCE CODE**

tolower("PRIYA")

**OUTPUT**

> tolower("PRIYA")

[1] "priya"

**(6)paste(string,sep=””):**

**SOURCE CODE**

paste("priya",3:4,sep="")

**OUTPUT**

> paste("priya",1:3,sep="")

[1] "priya1" "priya2" "priya3"

**(7)** **grep(pattern, x , ignore.case=FALSE, fixed=FALSE)**

**SOURCE CODE**

grep("A", c("b", "C", "A"), fixed = TRUE)

**OUTPUT**

> grep("A", c("b", "C", "A"), fixed = TRUE)

[1] 3

**3.STATISTICAL FUNCTIONS**

**(1)MEAN:**

**SOURCE CODE**

# Create a vector.

x <- c(10,10,10,10)

# Find Mean.

result.mean <- mean(x)

print(result.mean)

**OUTPUT**

> # Create a vector.

> x <- c(10,10,10,10)

>

> # Find Mean.

> result.mean <- mean(x)

> print(result.mean)

[1] 10

**(2)MEDIAN:**

**SOURCE CODE**

# Create the vector.

x <- c(1,2,3,4,5,6)

# Find the median.

median.result <- median(x)

print(median.result)

**OUTPUT**

> # Create the vector.

> x <- c(1,2,3,4,5,6)

>

> # Find the median.

> median.result <- median(x)

> print(median.result)

[1] 3.5

**(3)MODE:**

**SOURCE CODE**

# Create the function.

getmode <- function(v) {

uniqv <- unique(v)

uniqv[which.max(tabulate(match(v, uniqv)))]

# Create the vector with numbers.

v <- c(2,1,2,3,1,2,3,4,1,5,5,3,2,3)

# Calculate the mode using the user function.

result <- getmode(v)

print(result)

**OUTPUT**

> # Create the function.

> getmode <- function(v) {

+ uniqv <- unique(v)

+ uniqv[which.max(tabulate(match(v, uniqv)))]

+ }

>

> # Create the vector with numbers.

> v <- c(2,1,2,3,1,2,3,4,1,5,5,3,2,3)

>

> # Calculate the mode using the user function.

> result <- getmode(v)

> print(result)

[1] 2

**(4)LINEAR REGRESSION:**

### 1. Create Relationship Model & get the Coefficients

**SOURCE CODE**

print(relation)

x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.relation <- lm(y~x)

print(relation)

**OUTPUT**

> print(relation)

Call:

lm(formula = y ~ x)

Coefficients:

(Intercept) x

-38.4551 0.6746

### 2.Get the Summary of the Relationship

# Apply the lm() function.

relation <- lm(y~x)

print(summary(relation))

**OUTPUT**

> # Apply the lm() function.

> relation <- lm(y~x)

>

> print(summary(relation))

Call:

lm(formula = y ~ x)

Residuals:

Min 1Q Median 3Q Max

-6.3002 -1.6629 0.0412 1.8944 3.9775

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -38.45509 8.04901 -4.778 0.00139 \*\*

x 0.67461 0.05191 12.997 1.16e-06 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.253 on 8 degrees of freedom

Multiple R-squared: 0.9548, Adjusted R-squared: 0.9491

F-statistic: 168.9 on 1 and 8 DF, p-value: 1.164e-06

### 

### 3.Predict the weight of new persons

### 

### SOURCE CODE

# The predictor vector.

x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

# The resposne vector.

y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.

relation <- lm(y~x)

# Find weight of a person with height 170.

a <- data.frame(x = 170)

result <- predict(relation,a)

print(result)

**OUTPUT**

> # The predictor vector.

> x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

>

> # The resposne vector.

> y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

>

> # Apply the lm() function.

> relation <- lm(y~x)

>

> # Find weight of a person with height 170.

> a <- data.frame(x = 170)

> result <- predict(relation,a)

> print(result)

1

76.22869

|  |  |
| --- | --- |
| **EX.NO:2** | **GETTING INPUTS THROUGH KEYBOARD** |
|  |

mydata <- data.frame(name=character(0),age=numeric(0),

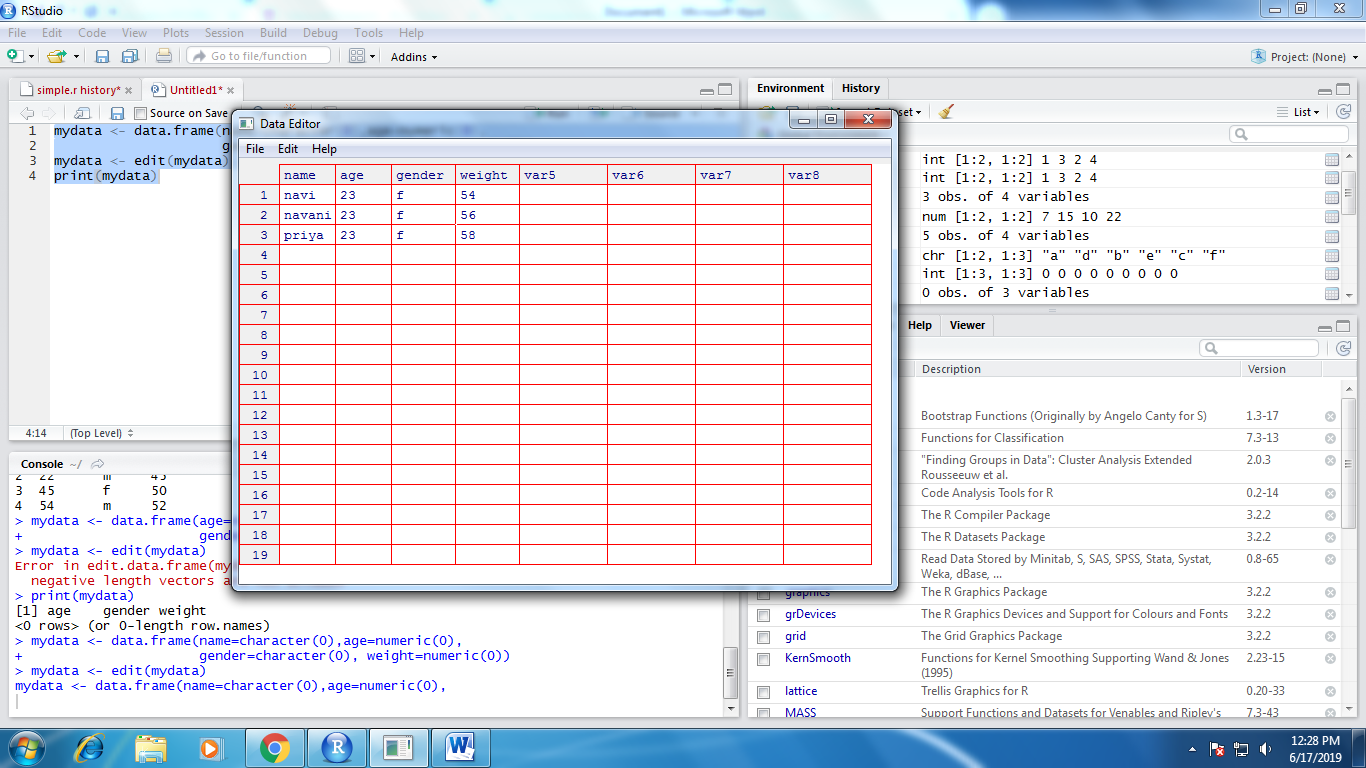
gender=character(0), weight=numeric(0))

mydata <- edit(mydata)

print(mydata)

**OUTPUT:**

**Giving inputs through keyboard:**



**Output on the console:**

Print(mydata)

Name age gender weight

1 navi 23 f 54

2 navani 23 f 56

3 priya 23 f 58

**URL:** <https://www.tutorialspoint.com/r/r_variables.htm>

|  |  |
| --- | --- |
| EX.NO:3 | **CREATING VARIABLES,RENAMING VARIABLES,MISSING VALUES,TYPE CONVERSION,SORTING DATA,MERGING DATASETS,SUBSETTING DATASETS** |
|  |

**1.CREATING NEW VARIABLES:**

**SOURCE CODE(1)**

# Assignment using equal operator.

var.1 = c(0,1,2,3)

**OUTPUT**

> # Assignment using equal operator.

> var.1 = c(0,1,2,3)

**SOURCE CODE(2)**

# Assignment using leftward operator.

var.2 <- c("learn","R")

cat ("var.2 is ", var.2 ,"\n")

**OUTPUT**

|  |
| --- |
| > # Assignment using leftward operator.  > var.2 <- c("learn","R")  > cat ("var.2 is ", var.2 ,"\n")  var.2 is learn R |
|  |
| |  | | --- | |  | |

**SOURCE CODE(3)**

# Assignment using rightward operator.

c(TRUE,1) -> var.3

cat ("var.3 is ", var.3 ,"\n")

**OUTPUT**

> # Assignment using rightward operator.

> c(TRUE,1) -> var.3

> cat ("var.3 is ", var.3 ,"\n")

var.3 is 1 1

**2.RENAMING VARIABLES:**

**SOURCE CODE**

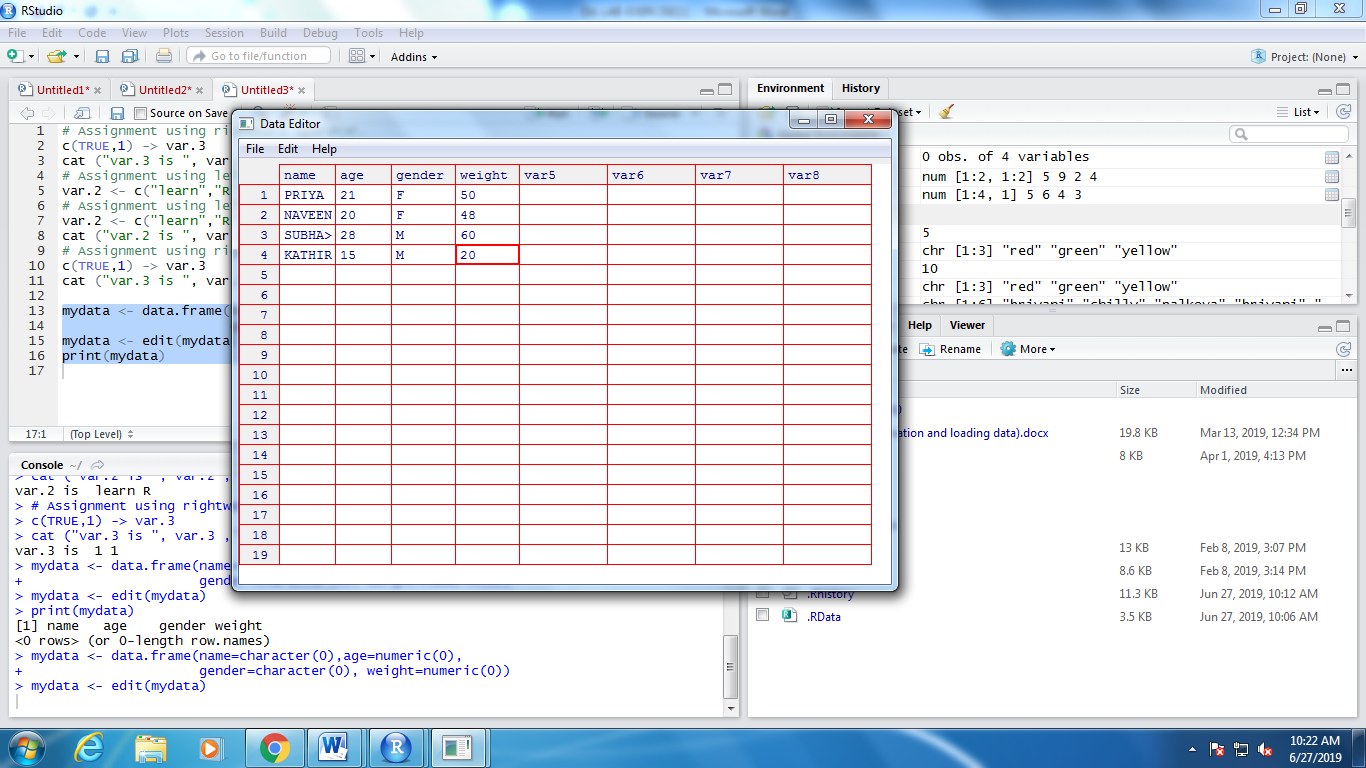
mydata <- data.frame(name=character(0),age=numeric(0),

gender=character(0), weight=numeric(0))

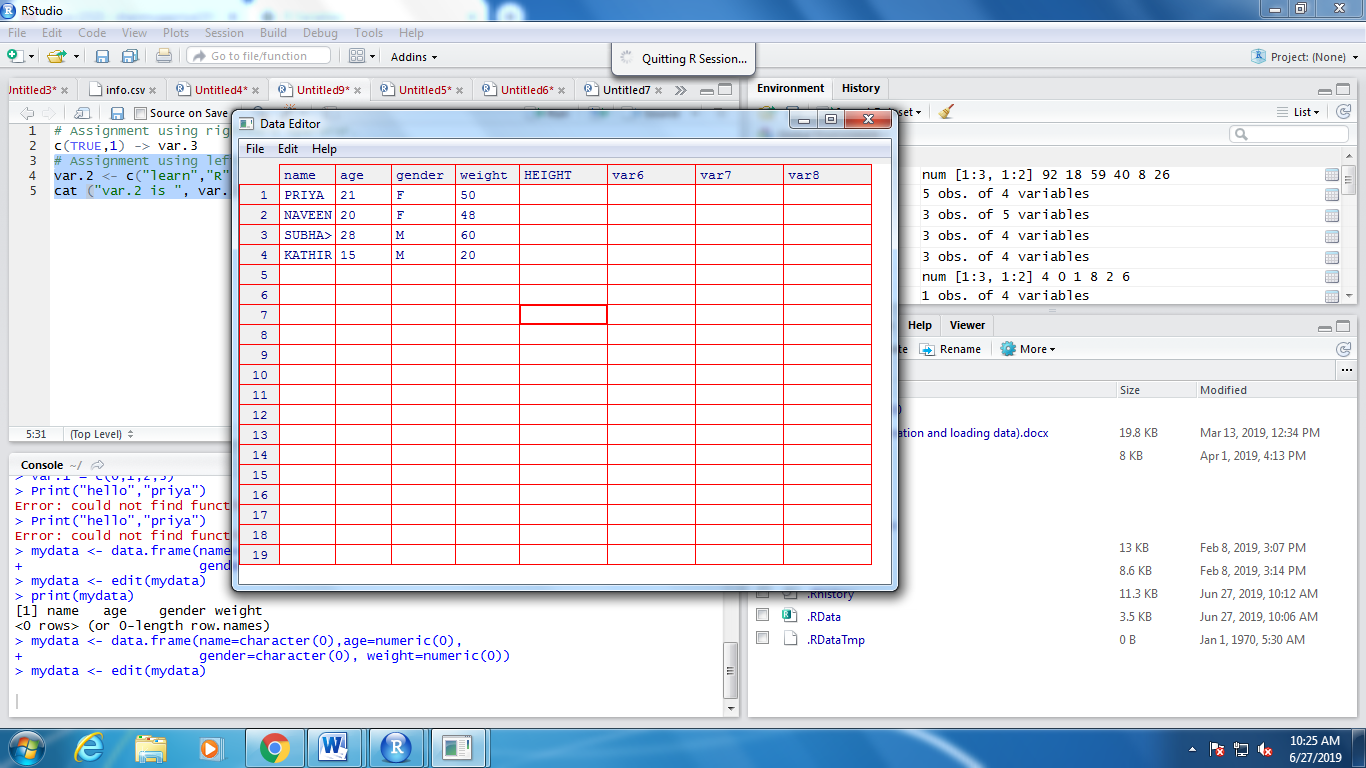
mydata <- edit(mydata)

print(mydata)

**OUTPUT SCREEN**



**CHANGING THE VARIABLE VAR5 AS HEIGHT**



**SORTING THE DATA**

**SOURCE CODE**

a <- c(50000, 1000, 10000)

order(a)

a[order(a**)]**

**OUTPUT**

> a <- c(50000, 1000, 10000)

> order(a)

[1] 2 3 1

> a[order(a)]

[1] 1000 10000 50000

**MERGING THE DATASET**

**SOURCE CODE**

X <- c(7,12,28,3,41)

Y <- c(8,9,10,11,12)

Z <- cbind(X,Y)

print(Z)

**OUTPUT**

> X <- c(7,12,28,3,41)

> Y <- c(8,9,10,11,12)

> Z <- cbind(X,Y)

> print(Z)

X Y

[1,] 7 8

[2,] 12 9

[3,] 28 10

[4,] 3 11

[5,] 41 12