**Day 01**

c(90,88,85)+ c(90,90,80)

num1<- c(90,88,85)

num2<- c(90,90,80)

num1+num2

string1<- "Welcome to Data Science"

string1

string2<- 'Welcome to Data Science'

string2

pi

exp(1)

log(exp(1))

log10(1000)

Y<-seq(1:10)

Y<-seq(1,10,2)

Y<-seq(1,10,0.5)

Y

rep("A", times= 10)

rep(1,times=3)

?seq

x<-c(1,3,5,7,8)

rep(x,times=3)

rep(x,each=3)

x<-c(1,3,5,7,8)

c(x,rep(seq(1,9,2),c(1,2,3),42,2:4))

v1<-c(2,5,7,9)

c(v1,rep(seq(2,9,2),c(1,2,3),42,2:4))

**Day 02**

**######Evaluation#########**

**#When a complete expression is entered at the prompt, it is evaluated and the result of the evaluated expression is returned. The result may be auto-printed.**

x <-1

x

msg <- "hello"

msg

**#R as a Calculator ----**

1

1+49.5

1-49

1+1/5\*6-3.4

**#Addition**

3+2

**#Subtraction**

3-2

**#Multiplication**

3\*2

3/2

log(10,10)

log(100,10)

3^2

3^(-3)

100^(1/2)

sqrt(100)

**# Vectors and Assignment ----**

c(1,2,3)

name<- c("Mark", "Zaid", "Jehan")

name

**#By using the assignment operator to assign the vector to a variable.**

Numbers<-c(1,2,3)

Numbers

People<-c("Ali", "bet", "cat")

People

x <- 4

x <- 5

y <- 6

Z1 <- x+y

Z1

z2 <- x-y

z3 <- x\*y

z4 <- x/y

**#Vectors and Assignment**

**#Simple arithmetic operations can be performed with vectors.**

c(1,2,3)+c(4,5,6)

Numbers1<-c(1,2,3)

Numbers1

sum(Numbers1)

mean(Numbers1)

length(Numbers1)

c(1,2,3) + c(1,2,3)

Numbers1+Numbers1

Numbers2<-c(8,7.5,-2)

Numbers2

Y<-c(1,2,3)

X<-c(1,2,3)

Z<-Y\*X

Z

V1<-c(1,2,3)

V2<-c(12,12,12)

V3<-V2+V1

V3

V2<-c(12,12,12,6)

V4<-c(12,12,12,12)

V5 <- V4/V2

V5

**#The outcome of an arithmetic calculation can be given as an identifier for later use.**

A<-c(8,7.5,-2)

cal1<-Z+A

cal1

cal2<-cal1\*cal1

cal2

V1 <- c(1, 3, 5, 7, 8, 9)

V2<-4\*V1

V2

V1

**#When an R vector is printed you will notice that an index for the vector is printed in square brackets [] on the side. For example, see this integer sequence of length 20.**

y <- 1:25

y

?seq

seq(from = 1.5, to = 4.2, by = 0.1)

seq(1.5, 4.2, 0.1)

Y<-seq(1,10)

Y

Y<-seq(1,10,2)

Y

Y<-seq(1,10,0.5)

Y

Y<-seq(1,100,4)

Y

rep("A", times = 10)

V1 <- c(1, 3, 5, 7, 8, 9)

rep(V1, times = 3)

c(V1, rep(seq(1, 9, 2), 3), c(1, 2, 3), 42, 2:4)

**#Creating Vectors**

**#The c() function can be used to create vectors of objects by concatenating things together**.

**# numeric**

x <- c(0.5, 0.6)

**# logical**

x <- c(TRUE, FALSE)

**# logical**

x <- c(T, F)

**# character**

x <- c("a", "b", "c")

**# integer**

x <- 9:29

**# complex**

x <- c(1+0i, 2+4i)

**#Explicit Coercion**

**Objects can be explicitly coerced from one class to another using the as.\* functions, if available.**

x <- 0:6

x

class(x)

as.numeric(x)

as.character(x)

x <- c("a", "b", "c")

as.numeric(x)

**#Matrices**

m <- matrix(1:6, nrow = 3, ncol = 2)

m

**#Matrices can also be created directly from**

**vectors by adding a dimension attribute.**

m <- 1:10

m

dim(m) <- c(2, 5)

m

**#Matricescanbecreatedbycolumn-binding orrow-binding withthecbind () andrbind () functions.**

x <- 1:3

x

y <- 10:12

y

cbind(x, y)

rbind(x, y)

**#Missing Values**

**##Create a vector with NAs in it**

x <- c(1, 2, NA, 10, 3)

x

**## Return a logical vector indicating which elements are NA**

**is.na(x)**

**#DataFrames**

**#R Object**

**The concatenation function can be used to concatenate vectors**.

a<-c(0,1)

b<-c(0,0)

c(a,b,a)

**Day 03**

**#data frame**

emp\_id<-c (101:105)

emp\_name<-c("bala","Silva","Perera","Nihashie", "Dulari")

emp\_details<- data.frame(emp\_id,emp\_name)

emp\_details

emp\_details$Salary<- c(50000,60000,55000,61000,70000)

emp\_details

**#structure of the data frame.**

str(emp\_details)

**#summary of salary**

summary(emp\_details$Salary)

**#average**

mean(emp\_details$Salary)

**#median**

median(emp\_details$Salary)

**#IQR**

IQR(emp\_details$Salary)

**#standard deviation**

sd(emp\_details$Salary)

round(sd(emp\_details$Salary),2)

**#extracting columns for employee number and names.**

result<- data.frame(emp\_details$emp\_id,emp\_details$emp\_name)

result

result<- data.frame(emp\_details['emp\_id'],emp\_details['emp\_name'])

result

**#extracting the first 2 rows**

result1<- emp\_details[1:2]

result1

result2<- emp\_details[1:2,1:3]

**#[row,columns]**

result2

**#extracting the 3rd and 5th row with 1st and 3rd column**

result3<- emp\_details[c(3,5),c(1,3)]

result3

row6<-list(106,"Kumara",610000)

row6

result4 <- rbind(emp\_details,row6)

result4

**#adding a new variable with data**

result4

result4$gender<-c("M","F","M","F","F","")

result4

result4

result4$gender<-c("M","F","M","F","F","M")

result4

example\_data<-data.frame(x=c(1,3,5,7,9,1,3,5,7,9),

y=c(rep("Hello",9),"Goodbye"),

Z=rep(c(TRUE,FALSE),5))

example\_data

**#checking the current working directory**

getwd()

**# setting a new working directory.**

setwd("/Users/dinilranditha/Desktop/untitled folder")

**#csv**

?read.csv

data3<-read.csv("employee-data.csv",header=T)

data3

**#view the data set in a new window**

View(data3)

**#to obtain variable names**

names(data3)

?read.csv

data4<-read.csv("employee-data1.csv",header=T)

data4

View(data4)

names(data4)

**#identfying the variables and ibservations.(rows and columns)**

dim(data4)

**#to obtain the first 6 rows of the data set, head function can be used**.

head(data4)

**# to obtain the last 6 rows.**

tail(data4)

**#structure of the data set**

str(data4)

**#subset (keeping row 5 to 10)**

data4[c(5,6,7,8,9,10),]

data4[c(5:10),]

data4[-(11:989),]

**#finding the variable type**

class(data4$emp\_no)

class(data4$birth\_date)

class(data4$gender)

summary(data4)

summary(data4$gender)

data4$gender<-as.factor(data4$gender)

class(data4$gender)

summary(data4$gender)

class(data4$title)

data4$title<-as.factor(data4$title)

class(data4$title)

summary(data4$title)

**#convert column to numeric**

class(data4$salary)

data4$salary<-as.numeric(data4$salary)

summary(data4$salary)

class(data4$salary)

**#Install packages**

install.packages("tidyverse")

library(tidyverse)

install.packages("devtools")

devtools::install\_github("r-lib/conflicted")

library(conflicted)

library(dplyr)

filter(mtcars, cyl == 8)

**R Practical**

**#1**

x <- c (90, 50, 70, 80, 70, 60, 20, 30, 80, 90, 20)

mean(x)

median(x)

sort(x)

sum(x)

length(x)

**#2**

y <- c (62000, 64000, 49000, 324000, 1264000, 54330, 64000, 51000,

55000, 48000, 53000)

mean(y)

median(y)

**#warmup quiz**

**#1**

Marks <- c(5, 15, 25, 35, 45, 55)

Students <- c(10, 20, 30, 50, 40, 30)

submarks <- data.frame(Marks, Students)

submarks

mean(Marks)

median(Marks)

mode(Marks)

**#bala quiz**

price <- c(64, 59, 72, 69, 55, 72, 60, 48, 75, 72, 81, 54)

mean(price)

**#round function of mean value**

round(mean(price),2)

median(price) **#Q2**

var(price) **# vichalanaya**

round(var(price),3)

min(price)

max(price)

summary(price)

IQR(price)

**#quartile**

quartiles <- quantile(price, probs = c(0.25 , 0.5, 0.75, 0.9))

quartiles

**#percentile**

percentile <- quantile(price, probs = c(0.1, 0.2, 0.9))

sort(price)

percentile

sum(price)

length(price)

sort(price, decreasing = F)

sort(price, decreasing = T)

**# we can use true and false insted of TandF**

?sort

?mean

**Day 04**

**(Jupyter Notebook)**

x = 'Malith'

print(x.capitalize())

print(x.upper())

print(x.lower())

print(x.casefold())

print(x[0])

print(x[3])

print(x[1:4])

print(x[1:])

print(x[:3])

print(x.replace('n', 'm'))

print(x.split('U'))

print(x.split('u'))

y = 'Sasindu'

print(y.split('a'))

print(y.split())

a = y.split(' ')

b = ' and '.join(a)

print(b)

n=12

m=16

print(x==y)

print(x!=y)

print(n<m)

print(n>m)

print(n<=m)

o = 12

print(o <= n)

print(o <= n)

print(o >= m)

print(m >= o)

aa = 'malith'

ab = 'MALTH'

ac = "maLIth"

print(ac.isupper())

print(aa)

print(aa.islower())

print(ac.islower())

print(ab.islower())

print(x)

print(3>1 or 6>3)

print(3<1 or 6>3)

print(3<1 or 3>6)

print(x)

print(not 3>1)

print(not 1>3)

nn=2

**#IF else**

if nn>50:

print('pass')

else:

print('fail')

mm=0

if mm>0:

print('fuck')

elif mm==0:

print('fuck off')

else:

print('ligma')

dd=-4

if dd>0:

print('balla')

else:

if dd<0:

print('kukula')

else:

print('paraviya')

**#LIST**

aaa=[1,2,3,4,5]

bbb=['a','b','c','d']

ccc = [5, 4, 3, 2, 1, 'f', 'python', [6, 7, 8], True, False]

print(ccc[5])

print(bbb[0])

print(ccc[6])

print(ccc[7])

aaa[3]=9

print(aaa)

print(len(ccc))

print(ccc)

ccc.insert(6,'banduwathi')

print(ccc)

ccc.remove('banduwathi')

print(ccc)

del aaa

print(ccc)

ccc.clear()

print(ccc)

aaa = [6,88,34,74,23,7,3,200]

aaa.sort()

print(aaa)

aaa.sort(reverse=True)

print(aaa)

del aaa

a,b,\*c, = [1,2,3,4]

print(a)

print(b)

print(c)

x=(1,2,3,4)

del x

del y

x=()

y=('java')

z=("hava",)

print(x)

print(y)

print(z)

import timeit

list\_time = timeit.timeit(stmt="[1,2,3,4]")

tuple\_time = timeit.timeit(stmt="(1,2,3,4)")

print(list\_time\*1000)

print(tuple\_time\*1000)

**#end first 10 video**