#R as calculator

10 + 20

30 \* 40

2 + 3 \* 5 \* (40-35)

# R comments

#We can add comments/ documentation in the program

#using the # symbol at the beginning of the line

# R help

help()

help("vector")

help.search("vector")

help.start()

find("max")

find("svm")

# Assignment <-, ->, =

X <- 10

x <- 100

Y = 20

30 -> Z

v1  <- c(1,2,3,4,5)

v2 <- c(56,45,78,89,34)

c(43,765,342) -> v3

a <- 200

A <- 500

# Arithmetic operators (+,-,\*,/,^,%% (modulus),%/% (integer division))

# demo for single numbers and vector

2 ^ 3

v1 ^ 2

v1

v1 \* 2

v1 / 2

v1 %% 2

v1 %/% 2

v3 <- c(100,200,300,400,500)

v4 <- c(2,3,4,5,6)

print(v3)

print(v4)

v3 + v4

v3 - v4

v3 \* v4

v3 / v4

v3 %% v4

v3 %/% v4

10 %% 2

10 %% 3

456 %% 2

567 %% 2

v3^v4

v3 <- c(10,11,12,13,14)

v3^v4

v3 <- c(100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,1400,1500)

v4 <- c(2,3,4,5,6)

v3 + v4

v3 <- c(100,200,300,400,500,600,700,800,900,1000,1100,1200)

v4 <- c(2,3,4,5)

v3 + v4

v3 <- c(100,200,300,400,500,600,700,800,900,1000,1100,1200,1300)

v4 <- c(2,3,4,5)

v3 + v4

v3 <- c(100,200,300)

v4 <- c(2,3,4,5)

v3 + v4

#Comparison/Relational operators (==,!=,<,>,<=,>=)

10 > 5

10 < 5

v4 <- c(10,34,67,45,94)

v5 <- c(11,55,67,8,91)

v4 < v5

v4 > v5

v4 == v5

v4 <= v5

v4 >= v5

v4 != v5

v6 <- c(10,NA,0,45,6+3i,5+4i,33+2i,67.8)

v7 <- c(11,55,67,8,91,8+10i,33+2i,67.8)

v6 == v7

# Logical operators (&,&&,|,||,!)

# & - Elementwise 'AND', | - Elementwise 'OR'

# &&, || - Only first element is checked

b1 <- TRUE

b2 <- FALSE

b1 && b2

b1 || b2

b1 & b2

b1 | b2

b3 <- c(FALSE,FALSE,TRUE,TRUE)

b4 <- c(FALSE,TRUE,FALSE,TRUE)

b3 && b4

b3 || b4

b3 & b4

b3 | b4

v6 <- c(10,NA,0,45,6+3i,5+4i,33+2i,6,0)

v7 <- c(11,55,67,8,91,NA,33+2i,0,0)

v6 & v7

v6 | v7

# Variables

# Characters, numbers, dot are allowed to use in variable names

# Variable name can start with letter, dot

# Variable name cannot start with underscore

# Variable name cannot include any special character

(76.8+98.5+87.9)/3

var1 <- 100

Var1 <- 200

vAR1 <- 300

var.value <- 500

.k <-60

# print function

print("This is SIBAR MBA")

print("MBA Sem II has BBAUR in syllabus")

print("There are "," 5 students in the class")

print("Value of var1 is: " + var1)

# cat function

cat("There are "," 5 students in the class")

cat("Value of var1 is: ", var1)

cat("There are three specializations available:","\n","BA","\n","Finance","\n","Marketing")

#  ------ Data Types ------ #

# logical, numeric, integer, complex, character, raw

# class function can be used for finding the data type

print(charToRaw("Hello SIBAR"))

print(charToRaw("Madan Vikram Shivani GAURI SAniYA"))

bool\_var = TRUE

class(bool\_var)

bool\_var2 = FALSE

class(bool\_var2)

num\_var = 345.67

class(num\_var)

num\_var2 = 50

class(num\_var2)

print(num\_var2)

num\_var3 <- c(50, 55.89)

class(num\_var3)

print(num\_var3)

int\_var = 100L

class(int\_var)

int\_var2 = 100.67L

class(int\_var2)

comp\_var = 37 + 8i

class(comp\_var)

char\_var = "A"

class(char\_var)

char\_var2 = "This is R Lab Session"

class(char\_var2)

raw\_var = print(charToRaw("Hello SIBAR"))

class(raw\_var)

print(rawToChar(raw\_var))

# ----- Data Structures ------#

# Vectors, Lists, Dataframes, Matrices, Arrays, Factors

vec1 <- c(10,20,30)

print(vec1)

vec2 <- c(2.5,5.6,3.5,9)

print(vec2)

vec3 <- c("This","is","R","Session")

print(vec3)

vec4 <- c(67,89,45.3,23.1,"Hello All")

print(vec4)

vec5 <- print(as.integer(vec2))

vec5

print(vec4[1])

if(23.1 %in% vec4)

  print("Yes")

l = length(vec4)

l

for(i in 1:l)

{

  if(vec4[i]==23.1)

    cat("Position of 23.1 is ",i)

}

vec6 <- c(67,23.1,89,45.3,23.1,"Hello All",23.1,"Madan","Vikram",67)

print(vec6)

l2 <- length(vec6)

print(l2)

for(i in 1:l2)

{

  if(vec6[i]==23.1)

    cat("Position of 23.1 is ",i,"\n")

}

for(i in 1:l2)

{

  print(i)

  if(vec6[i]==23.1)

    cat("Position of 23.1 is ",i,"\n")

}

for(i in 1:l2)

{

  print(i)

  if(vec6[i]==23.1)

    print("Position of 23.1 is ")

    print(i)

}

#matrix

#Syntax is matrix(data,nrow,ncol,byrow)

#By default byrow is FALSE

mat1 <- matrix(c(1,2,3,4,5,6,7,8),2,4)

print(mat1)

print(nrow(mat1))

print(ncol(mat1))

print(dimnames(mat1))

####dimnames(mat1)<-list(list('r1','r2'),list('c1','c2','c3'))

#####list(c('r1','r2'),c('c1','c2','c3'))

mat2 <- matrix(c(1,2,3,4,5,6,7,8),4,2)

print(mat2)

mat3 <- matrix(c(1,2,3,4,5,6,7,8),2,4,byrow=TRUE)

print(mat3)

mat4 <- matrix(c(1,2,3,4,5,6,7,8),4,2,byrow=TRUE)

print(mat4)

mat5 <- matrix(c(1:24),6,4)

mat5

mat6 <- matrix(c(1:24),6,4,byrow=TRUE)

mat6

mat7 <- matrix(c(1:24),6,7,byrow=TRUE)

mat7

mat4

mat8 <- t(mat4)

mat8

print(mat4 \* 5)

print(mat4 + 5)

print(mat4 - 5)

mat9 <- matrix(c(1,3,2,2,1,1),3,2,byrow=TRUE)

mat10 <- matrix(c(1,2,3,4,5,6),2,3,byrow=TRUE)

mat11 <- matrix(c(1,2,3,4,5,6,7,8,9),3,3,byrow=TRUE)

mat9

mat10

mat11

print(mat9%\*%mat10)

print(mat9%\*%mat11)

print(mat11%\*%mat9)

mat3

t\_mat3 <- t(mat3)

t\_mat3

mat3 %\*% t\_mat3

t\_mat3 %\*% mat3

cross <- crossprod(mat3)

print(cross)

print(cross[1,2])

print(cross[3,4])

print(cross[,3])

print(cross[2,])

print(cross[1:3,1:3])

print(cross[1:2,3:4])

print(cross[2:3,])

#array

# Syntax is array(data,dim=c())

arr1 <- array(c(34,23,65,79,31,98,76,73,49,41,83),dim=c(12,1,1))

arr1

arr2 <- array(c(34,23,65,79,31,98,76,73,49,41,83),dim=c(6,2,1))

arr2

arr3 <- array(c(34,23,65,79,31,98,76,73,49,41,83),dim=c(3,2,2))

arr3

data <- c(34,23,65,79,31,98,76,73,49,41,83)

arr4 <- array(data,dim=c(3,2,4))

arr4

arr5 <- array(c(1:60),dim=c(3,4,5))

arr5

arr6 <- array(c(1:30),dim=c(3,4,5))

arr6

# List

# list (collection of data structures)

list1 <- list(1,2,3,4,5)

list1

list2 <- list(c(1,2,3,4,5),c(10,20,30))

list2

list3 <- list(c(1,2,3,4,5),c(10,20,30),arr1,mat1)

list3

list3

print(list3[[1]][1])

print(list3[[1]][4])

print(list3[[2]][2])

print(list3[[3]][1])

print(list3[[3]][11])

print(list3[[4]][1])

print(list3[[4]][2])

print(list3[[4]][3])

print(list3[[4]][4])

print(list3[[4]][5])

print(list3[[4]][6])

print(list3[[4]][7])

print(list3[[4]][8])

list4 <- list(c(1,2,3,4,5),c(10,20,30),arr1,mat1,df1)

list4

#data frame

roll\_no <- c(1,2,3,4,5)

name <- c("Ramesh","Suresh","Ganesh","Mahesh","Rajesh")

course <-c("MCA","MCA","MCA","MBA","MBA")

year <- c(1,1,2,1,2)

city <- c("Pune","Mumbai","Pune","Delhi","Nagpur")

df1 <- data.frame(roll\_no,name,course,year,city)

df1

View(df1)

df2 <- data.frame(Sr.No=c(1,2,3,4,5),

                  Flavor = c("Vanilla","Strawberry","Mango","Kesar","Pista"),

                  Price = c(30,40,45,35,30))

print(df2)

print(df[1,2])

print(df2[3,2])

print(df2[1:2,2:3])

print(df2[2:4,1:3])

print(df2[2:4,2:3])

#factors

# Factors

# Used for categorise the data

# Syntax is factor(x = character(), levels,

#labels = levels,

#exclude = NA, ordered = is.ordered(x), nmax = NA)

df3 <- data.frame(rollno,name,course,city,per)

df3

str(df3)

df3$name

course <- as.factor(course)

str(course)

df3 <- data.frame(rollno,name,course,city,per)

df3

str(df3)

table(course)

table(name)

table(city)

install.packages("e1071")

installed.packages()

library(MASS)

attach(cats)

cats

View(cats)

str(cats)

data <- cats

factor\_levels <- c('F','M')

table(cats)

factor\_labels <- c(1,2)

cat\_gender\_factor <- factor(data$Sex, levels <- factor\_levels, labels <- factor\_labels)

cat\_gender\_factor

table(cat\_gender\_factor)

View(cats)

attach(iris)

View(iris)

iris

attach(AirPassengers)

str(AirPassengers)

str(Orange)

View(Orange)

#### conditions and loops

# if condition

salary <- 50000

cat("Old Salary: ", salary, "\n")

if (salary == 50000) { salary = salary \* 1.10}

cat("New Salary: ", salary,"\n")

salary <- 50000

cat("Old Salary: ", salary, "\n")

if (salary > 50000) { salary = salary \* 1.10}

cat("New Salary: ", salary,"\n")

salary <- 50000

dept <- 'Mktg'

if(salary <= 50000 && dept == 'Mktg') {

  salary = salary +4000

}

print(salary)

if(salary <= 50000 && dept == 'Mktg') {

  salary = salary +4000

}

print(salary)

if(salary <= 50000 || dept == 'Mktg') {

  salary = salary +4000

}

print(salary)

### if else

## If a student scores 90 percent marks give 3 additional percent to him/her

##.else give 1 additional percent

per <- 99

if(per>=90) {

  per = per +3

  if(per > 100) {per = 100}

  cat("3 percent added", "\n New percentage is: ", per)} else

{

  per = per +1

  cat("1 percent added","\n New percentage is ",per)

}

### Print the class obtained by a student####

#### Case 1: Fix the percentage ####

#### Case 2: Use scan function to take the input from user ####

per <- 69.99

per <- scan()

if(per>=70)

{ print("Class obtained is Distinction")} else

  if(per>=60 && per <70)

  {print("Class obtained is First Class")} else

    if(per>=55 && per <60)

    {print("Class obtained is Higher Second Class")} else

      if(per>=50 && per <55)

      {print("Class obtained is Second Class")} else

        if(per>=40 && per <50)

        {print("Class obtained is Pass Class")} else

        {print("Student has Failed in Exam")}

### ifelse

v1 <- c(1,2,3,4,5,6,7,8,9,10)

ifelse(v1 %% 2 == 0,"Even","Odd")

ifelse(v1 %% 2 == 0,v1+100,v1+500)

ifelse(v1<5,v1\*1000,v1\*200)

ifelse(v1<5,v1\*1000,v1\*20)

mat1 <- matrix(1:30,nrow=6,ncol=5)

mat1

ifelse(mat1%%2 == 0,"Even","Odd")

ifelse(mat1 >15,1,0)

#for loop

for(i in 1:10) { print(i)}

#Print multiplication table of 2

num <- 2

for(i in 1:10) {

  print(num\*i)}

for(i in 1:10) {

  cat(num," \* ",i," = ", num\*i,"\n")}

for(r in 1:5) {

  for(c in 1:5) {

    cat(" \* ")

  }

  cat("\n")

  }

for(r in 1:5) {

  for(c in 1:5) {

    if(r>=c) {

      cat(" \* ")

    }

  }

  cat("\n")

}

#repeat loop

num <- 20

repeat{

  print(num)

  num <- num -1

  if (num<10) {

    break

  }

}

num <- 20

repeat{

  print(num)

  num <- num -1

  if (num<10) {

    break

  }

  print(num\*2)

}

# while loop

num <- 15

while(num<30) {

  print("SIBAR")

  print(num)

  num <- num + 2

}

#mean, median

data <- c(1,4,6,4,9,3,11,90,87,56)

mean(data)

median(data)

sum(data)

min(data)

max(data)

length(data)

avg <- sum(data)/length(data)

avg

range(data)

data\_na <- c(1,4,6,4,9,3,NA,11,90,87,56,NA)

data\_na

mean(data\_na)

median(data\_na)

sum(data\_na)

min(data\_na)

max(data\_na)

length(data\_na)

avg <- sum(data\_na)/length(data\_na)

avg

range(data\_na)

mean(data\_na,na.rm=TRUE)

median(data\_na,na.rm=TRUE)

sum(data\_na,na.rm=TRUE)

min(data\_na,na.rm=TRUE)

max(data\_na,na.rm=TRUE)

length(data\_na)

avg <- sum(data\_na,na.rm=TRUE)/length(data\_na)

avg

range(data\_na,na.rm=TRUE)

data

print(var(data))

print(sqrt(var(data)))

data\_na

var\_data\_na <- var(data\_na,na.rm = TRUE)

std\_dev\_data\_na <- sqrt(var\_data\_na)

print(var\_data\_na)

print(std\_dev\_data\_na)

getwd()

org\_data <- read.csv('organizations\_data.csv')

str(org\_data)

View(org\_data)

data\_without\_headr <- read.csv('organizations\_data.csv',header=FALSE)

data\_without\_headr

View(data\_without\_headr)

setwd('/Users/priyapande/Downloads')

getwd()

org\_data <- read.csv('organizations\_data.csv')

str(org\_data)

View(org\_data)

getwd()

org\_data <- read.csv('/Users/priyapande/organizations\_data.csv')

str(org\_data)

install.packages('readxl')

library(readxl)

iris\_data <- read\_excel("iris.xls")

View(iris\_data)

str(iris\_data)

iris\_data <- as.data.frame(iris\_data)

str(iris\_data)

iris\_data$Sepal\_Length

str(iris\_data)

str(x)

#simple plot

#index value is taken at x-axis

#data value is taken at y-axis

#

x<-c(10,20,35,42)

plot(x,type="p") #p is for point

plot(x,type="o") #o is for line and point both

plot(x,type="l") #l is for line only

x <- seq(0,2,by=0.2) #by is interval value

x

y <- x

y

y1 <- y^2

y1

y2 <- y^3

y2

plot(x,y,"o",pch=1,lty=1,main="My Plot",xlab="",ylab="",

     ylim=range(0,max(y)),cex=0.7,lwd=2)

plot(x,y,"o",pch=3,lty=4,main="My Plot",xlab="",ylab="",

     ylim=range(0,max(y)),cex=0.7,lwd=5)

plot(x,y,"o",pch=6,lty=5,main="My Plot",xlab="",ylab="",

     ylim=range(0,max(y)),cex=0.7,lwd=1)

plot(x,y,"o",pch=1,lty=1,main="My Plot",xlab="x label",ylab="y label",ylim=range(0,max(y2)),cex=0.7,lwd=2)

lines(x,y1,"o",pch=2,lwd=2) #adds line to the current plot

lines(x,y2,"o",pch=3,lwd=2)

x1<-seq(0.15,2,length=20)

x1

x2<-seq(0.4,2,length=15)

x2

y3<-x1^-1

y3

lines(x1,y3,"o",pch=4,lwd=2)

y4<-x2^-2

y4

lines(x2,y4,"o",pch=5,lwd=2)

legend(locator(1),

       legend=

         c("line","quadratic","cubic",

           "power=-1","power=-2"),

       pch=1:5,cex=0.8,lwd=2)

x<-c(2001,2002,2003,2004,2005)

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

plot(x,y,"o",pch=10,lty=3,main="My New Plot",xlab="Year",ylab="Courses",ylim=range(0,max(y)),cex=0.7,lwd=9)

 #\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#Merge the data frames

df1 <- data.frame(RNO = c("R1", "R2", "R3"), Names = c("Amit", "Sumit", "Namit"))

df2 <- data.frame(Names = c("Amit", "Sumit", "Namit"), City = c("Pune", "Mumbai", "Nashik"))

df1

df2

df <- merge(df1, df2, by.x ="Names", by.y ="Names")

print(df)

df1 <- data.frame(Names = c("Ramesh", "Suresh", "Naresh"), RNO = c(1,2,3))

df2 <- data.frame(RNO = c(1,2,3), City = c("Pune", "Mumbai", "Nashik"))

df1

df2

df <- merge(df1, df2, by.x ="RNO", by.y ="RNO")

print(df)

df1 <- data.frame(Names = c("Ramesh", "Suresh", "Naresh","Jayesh"), RNO = c(1,2,3,4))

df2 <- data.frame(RNO = c(1,2,3), City = c("Pune", "Mumbai", "Nashik"))

df1

df2

df <- merge(df1, df2, by.x ="RNO", by.y ="RNO")

print(df)

#SUBSET

library(MASS)

attach(painters)

print(painters)

View(painters)

is.data.frame(painters)

### as.data.frame(variable name) will convert the variable in data frame format

summary(painters)

summary(painters$Composition)

str(painters)

school <- print(painters$School)

print(painters$Composition)

median(painters$Composition)

sort(painters$Composition)

summary(School)

# finding subsets of the data

# subset(data\_name,condition)

View(painters)

print(School)

subset1 <- subset(painters,painters$School == 'A')

subset1

is.data.frame(subset1)

subset2 <- painters[painters[["School"]] == "F",]

subset2

subset3 <- subset(painters,painters$Drawing <= 10)

subset3

subset4 <- subset(painters,Expression == 0)

subset4

print(subset(painters,select=c(-1)))

subset5 <- subset(painters,School=="C",select=c(-2,-4))

subset5

# split (data\_name, factor\_column)

str(painters)

split1 <- split(painters,School)

split1

print(split1$A)

split1\_A <- split1$A

split1\_B <- split1$B

split1\_C <- split1$C

split1\_D <- split1$D

split1\_E <- split1$E

split1\_F <- split1$F

split1\_G <- split1$G

split1\_H <- split1$H

print(split1\_A)

print(split1\_B)

print(split1\_C)

print(split1\_D)

print(split1\_E)

print(split1\_F)

print(split1\_G)

print(split1\_H)

is.data.frame(split1\_A)

split2 <- split(painters,Composition>10)

split2

words <- c('Allana', 'AllanaInstitute', 'AllanaInstituteof',

           'InstituteofManagement', 'AllanaInstituteofManagementSciences','Institute')

words

grep('Allana', words)

grepl('Allana', words)

x <- c("Exam","will","include","examples","taken","in","internal exam")

x

grep("exam",x,value=T)

grep("exam",x,value=F)

grep("exam",x,value=T,ignore.case=T)

grep("exam",x,value=F,ignore.case=T)

a <- "R is an interesting language. Works of data"

b <- "R has interpreter. Data can be dealt easily"

c <- "R is used by data scientists"

#c(a,b,c)

grep("data",c(a,b,c),value=T,ignore.case=T)

grep("data",c(a,b,c),value=F,ignore.case=T)

grep("data",c(a,b,c),value=T,ignore.case = F)

grep("data",c(a,b,c),value=F,ignore.case = F)

df<-"R is a collaborative project with many contributors"

sub('R','R language',df)

df<-"R is a collaborative project with many contributors. I love R"

sub('R','R language',df)

gsub('R','R language',df)

df<-data.frame(Ele <- c('Flower','Goal','Hat','Gap','Cap','Cool','Hot','Horse'), Num <- c(1,2,3,4,5,6,7,8))

df

sub('G','A',df)

sub('H','C',df)

gsub('G','A',df)

gsub('H','C',df)

sub('G','A',df$Ele)

sub('H','C',df$Ele)

gsub('G','A',df$Ele)

gsub('H','C',df$Ele)

dfx <- "The boy is standing between boys surrounded with many boys"

sub("boy","donkey",dfx)

gsub("boy","donkey",dfx)

x <- "It rains heavily. Rain rains rainss rainsss"

gsub("rain","ice",x)

toupper(x)

tolower(x)

View(iris)

library(dplyr)

iris %>%

  group\_by(Species)  %>%

  summarize(Min\_Petal\_Width = min(Petal.Width))

iris %>%

  group\_by(Species)  %>%

  summarise(Count =n()) # Count All

iris %>%

  group\_by(Species)  %>%

  summarise(Count=n\_distinct(Petal.Width)) # Count Distinct

iris %>%

  group\_by(Species)  %>%

  summarise(Max\_Petal\_Width=max(Petal.Width))

iris %>%

  group\_by(Species)  %>%

  summarise(Avg\_Petal\_Width=mean(Petal.Width))

iris %>%

  group\_by(Species)  %>%

  summarise(Median\_Petal\_Width=median(Petal.Width))

iris %>%

  group\_by(Species)  %>%

  summarise(Median\_Petal\_Width = median(Petal.Width),

            Above\_Median\_Petal\_Width=median(Petal.Width[Petal.Width>Median\_Petal\_Width]))

iris %>%

  group\_by(Species)  %>%

  summarise(Sum\_Petal\_Width=sum(Petal.Width))

iris %>%

  group\_by(Species)  %>%

  summarise(Std\_Dev\_Petal\_Width=sd(Petal.Width))

iris %>%

  group\_by(Species)  %>%

  summarise(First = first(Petal.Width),Last=last(Petal.Width))

iris %>%

  group\_by(Species)  %>%

  summarise(Second = nth(Petal.Width,2))

iris %>%

  group\_by(Species,Petal.Width)  %>%

  summarise(Min\_Petal\_Width = min(Petal.Width)) %>%

  arrange(desc(Species,Petal.Width))

iris %>%

  group\_by(Species,Petal.Width)  %>%

  summarise(Min\_Petal\_Width = min(Petal.Width)) %>%

  arrange(order(Species,Petal.Width)) # Arrange in Ascending order

iris %>%

  filter(Species=='setosa') %>%

  group\_by(Species)  %>%

  summarise(Sum\_Petal\_Width=sum(Petal.Width))

iris %>%

  group\_by(Species)  %>%

  summarise(Sum\_Petal\_Width=sum(Petal.Width)) %>%

  ungroup() %>%

  summarise(Sum\_Petal\_Width=sum(Petal.Width))