SAVEETHA SCHOOL OF ENGINEERING

ITA0443

STATISTICS WITH R PROGRAMMING

LAB

keerthi

192120027

BI

BASIC OPERATIONS IN C

1.Write the Commands To Perform Basic Arithmetic In R.

Program:

a <- 19

b <- 4

add <- a + b

sub = a - b

multi = a \* b

division = a / b

Integer\_Division = a %/% b

exponent = a ^ b

modulus = a %% b

print(paste("Addition of a and b:", add))

print(paste("Subtracting Number a and b: ", sub))

print(paste("Multiplication of a and b: ", multi))

print(paste("Division of a and b: ", division))

print(paste("Integer Division of a and b: ", Integer\_Division))

print(paste("Exponent of a and b: ", exponent))

print(paste("Modulus of a and b: ", modulus))

Output:

> print(paste("Addition of a and b:", add))

[1] "Addition of a and b: 23"

> print(paste("Subtracting Number a and b: ", sub))

[1] "Subtracting Number a and b: 15"

> print(paste("Multiplication of a and b: ", multi))

[1] "Multiplication of a and b: 76"

> print(paste("Division of a and b: ", division))

[1] "Division of a and b: 4.75"

> print(paste("Integer Division of a and b: ", Integer\_Division))

[1] "Integer Division of a and b: 4 "

> print(paste("Exponent of a and b: ", exponent))

[1] "Exponent of a and b: 130321"

> print(paste("Modulus of a and b: ", modulus))

[1] "Modulus of a and b: 3"

2. Display a String on R Console.

To display a string on R Console by using a cat() and print() function.The cat() function combines multiple items into a continuous print output.

Program:

mes<-"HELLO WORLD!"

cat("R is Easy.\n",mes)

Output:

R is Easy.

HELLO WORLD!

3. Declare Variables In R And Also Write The Commands For Retrieving The Value Of The Stored Variables In R Console.

The variables can be assigned values using leftward, rightward and equal to operator. The values of the variables can be printed using print() and cat() function. The cat() function combines multiple items into a continuous print output.

Program:

# Assignment using equal operator.

var1= c(1,2,3,4)

# Assignment using leftward operator.

var2<- c("learn","R")

# Assignment using rightward operator.

c(2,1) -> var3

print(var1)

print(var2)

print(var3

Output:

> print(var1)

[1] 1 2 3 4

> print(var2)

[1] "learn" "R"

> print(var3)

[1] 2 1

4. Write R script to calculate the area of Rectangle.

Program:

height=5

width=6

area=height\*width

print(area)

Output:

[1] 30

5.Write Commands In R Console To Determine The Type Of Variable

We can use of the typeof() and class() function to check the datatype of a program.

Program:

a<-2

b<-"Set"

c<-TRUE

d<-as.integer(a)

typeof(a)

typeof(b)

typeof(c)

typeof(d)

Output:

> typeof(a)

[1] "double"

> typeof(b)

[1] "character"

> typeof(c)

[1] "logical"

> typeof(d)

[1] "integer"

6.Enumerate The Process To Check Whether A Given Input Is Numeric , Integer , Double, Complex in R.

Program: for integer

var=readline();

var=as.integer(var)

typeof(var)

Output:

> var=readline();

125

> var=as.integer(var)

> typeof(var)

[1] "integer"

For numeric

var=readline();

var=as.numeric(var)

typeof(var)

Output:

> var=readline();

2.5

> var=as.numeric(var)

> typeof(var)

[1] "double"

For complex

var=readline();

var=as.complex(var)

typeof(var)

Output:

> var=readline();

3+4i

> var=as.complex(var)

> typeof(var)

[1] "complex"

7. Illustration of Vector Arithmetic.

Program:

> vec1+vec2

[1] 4 7 7 6 11

> vec1-vec2

[1] -2 -3 -1 2 -1

> vec1\*vec2

[1] 3 10 12 8 30

> vec1/vec2

[1] 0.3333 0.4000 0.7500 2.0000 0.8333

8. Write an R Program to Take Input From User.

Input name as “Jack” and age as 17.

The program should display the output as

“Hai , Jack next year you will be 18 years old”

Program:

name <- readline(prompt="Enter name: ")

age <- readline(prompt="Enter age: ")

age <- as.integer(age)

print(paste("Hai,",name,"next year you will be",age+1,"years old."))

Output:

> name <- readline(prompt="Enter name: ")

Enter name: Jack

> age <- readline(prompt="Enter age: ")

Enter age: 17

> age <- as.integer(age)

> print(paste("Hai,",name,"next year you will be",age+1,"years old."))

[1] "Hai, Jack next year you will be 18 years old."

DATA STRUCTURES IN R

1. Perform Matrix Addition & Subtraction in R.

INPUT:

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

mymatrixa

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

mymatrixb

mymatrixadding<-mymatrixa+mymatrixb

mymatrixsubstraction<-mymatrixa-mymatrixb

OUTPUT:

> mymatrixa

[,1] [,2]

[1,] 1 4

[2,] 2 5

[3,] 3 6

> mymatrixb<-matrix(data=1:6,nrow=3,ncol=2)

> mymatrixb

[,1] [,2]

[1,] 1 4

[2,] 2 5

[3,] 3 6

Mymatrixadding int[1:3,1:2] 2 4 6 8 10 12

Mymatrixsubstraction int[1:3 , 1:2] 0 0 0 0 0 0

2)Perform Scalar multiplication and matrix multiplication in R

INPUT:

A <- matrix(1:4, 2, byrow = TRUE)

c <- matrix(rep(3, 4), 2)

A \* c

OUTPUT:

[,1] [,2]

[1,] 3 6

[2,] 9 12

MATRIX MULTIPLICATION

INPUT:

a = matrix(1:4,2,2)

3 \* a

c(1:2) %\*% a

c(1:2) %\*% t(a)

solve(a)

OUTPUT:

> c(1:2) %\*% a

[,1] [,2]

[1,] 5 11

> c(1:2) %\*% t(a)

[,1] [,2]

[1,] 7 10

> solve(a)

[,1] [,2]

[1,] -2 1.5

[2,] 1 -0.5

3)Find Transpose of matrix in R.

INPUT:

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

A <- matrix(data, nrow = 3, ncol = 2)

A\_T <- t(A)

print("Matrix A")

print(A)

print("Transpose of A")

print(A\_T)

OUTPUT:

[1] "Transpose of A"

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

[1,] 1 2 3

[2,] 4 5 6

4) Perform the operation of combining matrices in R using cbind().

functions.

INPUT:

a<-c(1,2,3)

b<-c(4,5,6)

c<-c(7,8,9)

m<-cbind(a,b,c)

print("the given 3\*3 matrix:")

print(m)

OUTPUT:

[1] "the given 3\*3 matrix:"

a b c

[1,] 1 4 7

[2,] 2 5 8

[3,] 3 6 9

5) Deconstruct a matrix in R.

INPUT:

B <-matrix(c(2, 4, 3, 1, 5, 7),nrow=3,ncol=2)

c(B)

OUTPUT:

[1] 2 4 3 1 5 7

6) Perform array manipulation in R .

INPUT:

vector1 <- c(1, 2, 3)

vector2 <- c(10, 15, 3, 11, 16, 12)

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

print(result)

OUTPUT:

, , 1

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

[1,] 1 10 11

[2,] 2 15 16

[3,] 3 3 12

, , 2

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

[1,] 1 10 11

[2,] 2 15 16

[3,] 3 3 12

7) Demonstrate Factor data structure in R.

INPUT:

X = c(1, 3, 5, 7, 8)

print(X)

OUTPUT:

[1] 1 3 5 7 8

8) Create a data frame and print the structure of the data frame in R.

INPUT:

Name <- c("harsha", "keerthi", "krisha", "sandy", "rahul")

Age <- c(22,19,17,18,21)

df <- data.frame(Name, Age)

print (df)

OUTPUT:

Name Age

1 harsha 22

2 keerthi 19

3 krisha 17

4 sandy 18

5 rahul 21

9) Demonstrate the creation of S3 class in R.

INPUT:

x<-c("geethika","rahul","preeti")

class(x)

output:

[1] "character"

10) Demonstrate the creation of S4 class in R.

Input:

setClass("Employee\_Info", slots=list(name="character", age="numeric", role="character"))

employee1 <- new("Employee\_Info", name = "rahul", age = 21, role = "HR")

employee1

OUTPUT:

An object of class "Employee\_Info"

Slot "name":

[1] "rahul"

Slot "age":

[1] 21

Slot "role":

[1] "HR"

WORKING WITH LOOPING AND FUNCTIONS IN R

1.Write a program to check whether an integer (entered by the user) is a prime number or not using control statements.

INPUT:

num<-as.integer(readline(prompt = "enter the number"))

fact=0

for (i in 3:num-1) {

if(num%%i==0)

{

print(i)

fact=1

}

}

if(fact==1)

{

print("not a prime")

}else

{

print("prime number")

}

OUTPUT:

enter the number15

[1] 3

[1] 5

[1] "not a prime”

2)Write a program to check whether a number entered by the user is positive number or a negative number or zero.

INPUT:

num<-as.integer(readline(prompt = "enter the number"))

if(num>0)

{

print("positive number")

}else if(num<0)

{

print("negative number")

}else

{

print("zero")

}

OUTPUT:

enter the number39

[1] "positive number"

3)Write a program to check whether a number is an Armstrong number or not using a while loop.

INPUT:

num = as.integer(readline(prompt="Enter a number: "))

sum = 0

temp = num

while(temp > 0) {

digit = temp %% 10

sum = sum + (digit ^ 3)

temp = floor(temp / 10)

}

if(num == sum) {

print(paste(num, "is an Armstrong number"))

} else {

print(paste(num, "is not an Armstrong number"))

}

OUTPUT:

Enter a number: 37

[1] "37 is not an Armstrong number"

4)Write a program to demonstrate Repeat Loop in R.

INPUT:

result <- c("Hello World")

i <- 1

repeat {

print(result)

i <- i + 1

if(i >5) {

break

}

}

OUTPUT:

[1] "Hello World"

[1] "Hello World"

[1] "Hello World"

[1] "Hello World"

[1] "Hello World"

5)Using functions develop a simple calculator in R.

INPUT:

add <- function(x, y) {

return(x + y)

}

subtract <- function(x, y) {

return(x - y)

}

multiply <- function(x, y) {

return(x \* y)

}

divide <- function(x, y) {

return(x / y)

}

# take input from the user

print("Select operation.")

print("1.Add")

print("2.Subtract")

print("3.Multiply")

print("4.Divide")

choice = as.integer(readline(prompt="Enter choice[1/2/3/4]: "))

num1 = as.integer(readline(prompt="Enter first number: "))

num2 = as.integer(readline(prompt="Enter second number: "))

operator <- switch(choice,"+","-","\*","/")

result <- switch(choice, add(num1, num2), subtract(num1, num2), multiply(num1, num2), divide(num1, num2))

print(paste(num1, operator, num2, "=", result))

OUTPUT:

[1] "Select operation."

[1] "1.Add"

[1] "2.Subtract"

[1] "3.Multiply"

[1] "4.Divide"

Enter choice[1/2/3/4]: 3

Enter first number: 22

Enter second number: 34

[1] "22 \* 34 = 748"

6) Demonstrate the creation of a complex number in R.

INPUT:

x <- 1

y <- 1

z <- complex(real = x, imaginary = y)

z

# [1] 1+1i

OUTPUT:

[1] 1+1i