1. Write a R program using control operators to test whether following values are prime numbers or not by providing a PRIME or NOT PRIME message as output :

A. 103 B. 82 C. 179

Sol: x = c(103, 82, 179)

for(z in x){

if((z %% 2 != 0) && (z %% 3 != 0)){

cat(z,": PRIME NUMBER \n")

}else{

cat(z, ": Not PRIME NUMBER \n")

}

}

Sol: 2

1. Write a R program using control operators to identify letter u and a both occur in the following words: 1. above 2. unit 3. Under

Sol: library(stringr)

library(dplyr)

words = c("above", "unit", "Under")

for(alphabet in words){

if((str\_detect(alphabet, "a"))==TRUE && ((str\_detect(alphabet, "u"))==TRUE)){

cat(alphabet, ": both u and a appear in each words \n")

}else{

cat(alphabet, ": both u and a doesn't' appear in each words \n")

}

}

1. Write a function that to calculate BMI (Body Mass Index): BMI for a person is defined as their body mass divided by the square of their height The weight is in kilograms and the height in meters or (The weight can be in pounds and the height in inches)\* 703

Sol: BMI <- function(weight, height){

BMI = (weight/(height^2))\*703

# weight in Kilogram and height in inches

return(BMI)

# 1kilogram = 2.20462262185, so weight(pound) = weight(kilogram)/ 0.45359237

}

BMI(50, 1.75)

1. Write a function called sum\_of\_cubes, that calculates the sum of cubes of the first n natural numbers : if we have two numbers : 1, 2 then sum of squares is 9 ( 1^3 + 2^3) if we have three numbers : 1, 2, 3 then sum of squares is 36 ( 1^3 + 2^3 + 3^3)

Sol: sum\_of\_cubes <- function(x){

cubes <- x^3

sum <- sum(cubes)

return(sum)

}

sum\_of\_cubes(6)

1. Write a function to calculate the mode (highest frequency) of the following vector: x = c(2,3,3,4,4,5,6,7,9,10)

Sol: get <- function(x){

Mode <- unique(x)

Mode[which.max(tabulate(match(x, Mode)))]

}

x = c(2,3,3,4,4,5,6,7,9,10)

get(x)

6) Write a function to calculate the no. of prime numbers of the following vector : x = c(2,2,3,3,4,5,7,11,15,19,24,29)

Sol: install.packages("numbers")

library(numbers)

x <- c(2,2,3,3,4,5,7,11,15,19,24,29)

calculate\_no\_primes <- function(x){

for(i in x){

if(isPrime(i)){

cat(i, ": Prime \n")

}else{

cat(i, ": Not Prime \n")

}

}

}

calculate\_no\_primes(x)

8) Perform below operations using Data.frame and Data.table

boys <- read.csv("boystop100s.csv", header = TRUE)

head(boys)

girls <- read.csv("girlstop100s.csv", header = TRUE)

head(girls)

stu <- data.frame(roll\_no = c(3,1,2,5,4), names = c('peter', 'jack', 'david', 'james', 'john'))

stu

library(data.table)

marks <- data.table(roll\_no = c(4,2,3,6,1), maths = c(89,92,76,67,90), science = c(98,92,88,91,92))

marks

Join <- merge(stu, marks)

Join

Join1 <- merge(stu, marks, by="roll\_no", all.x=TRUE)

Join1

Join2 <- merge(stu, marks, by="roll\_no", all.y=TRUE)

Join2

Join3 <- merge(stu, marks, by='roll\_no', all=TRUE)

Join3

fill <- Loblolly

fill

fill$height

library(dplyr)

filter(fill, !is.null(fill$height))[,2]

library(dplyr)

group\_by(Loblolly, age) %>% summarise(sum = sum(age),average = mean(age))

9) Create R functions for the following operations.

gas <- function(){

library(dplyr)

data("infert")

color <- infert

color

distinct(color, age)

}

gas()

10) Create R functions for the following operations

# (a): Find out if there are any nulls in a dataset or in some specific number of columns

# data(BOD)

mydata <- function(){

mydata <- BOD

mydata

if(is.null(mydata$demand) || (!complete.cases(mydata))){

print(TRUE)

}else{

print(FALSE)

}

}

mydata()

# Question (b) is out of context

11): Create R functions for the following operations

poise <- function(x){

viz <- x[duplicated(x)]

print(viz)

return(x)

}

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

poise(x)

Uni <- function(x){

sun <- unique(x)

return(sun)

}

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

Uni(x)

clip <- function(){

j <- "Planet"

w <- "World"

paste(j,w, sep="::")

}

clip()

Melt <- function(){

mat <- matrix(c(1:10), nrow=5,ncol=2)

cat("Sum column wise :", apply(mat, 2, sum), "\n")

cat("Sum row wise :", apply(mat, 1, sum))

}

Melt()

# Question e and f and out of context

12) Create R functions for the following operations

seat <- function(){

seat <- Seatbelts

seat

colnames(seat) <- c("driverkilled", "Drivers", "Front",

"Rear", "KM/S", "PetrolPrice",

"vankill", "LAW")

print(seat)

seat <- seat[,(names(seat)) %in% c("law", "vankill")]

print(seat)

x <- c(NA, NaN)

print(is.na(x))

print(is.nan(x))

print(class(NA))

print(class(NaN))

print(class(NULL))

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

if(is.numeric(vec)){

print(TRUE)

}

## g -- Compute number of unique combinations in a data frame grouped by certain columns

library(dplyr)

Orange

distinct(Orange, Tree)

}

seat()