# UNIVERSIDAD NACIONAL DE LOJA

**PERIODO ACADÉMICO: OCTUBRE 2019 MARZO 2020**

**PRACTICA # 4 ASIGNATURA: SIMULACIÓN**

**RESULTADO DE APRENDIZAJE DE LA PRÁCTICA: Entiende las estructuras básicas de programación en R**

**TIEMPO PLANIFICADO: 3 HORAS**

**NUMERO DE ESTUDIANTES: Sexto ciclo (Paralelo A)**

## TEMA: Programación básica en R

1. **OBJETIVOS:**
   * Comprende las estructuras básicas if, ifelse, for.
   * Comprende el uso de vectorización.
   * Usa los conocimientos aprendidos en teoría para su posterior aplicación práctica.

## RECURSOS NECESARIOS:

* + R-studio.
  + Computador de Laboratorios

## INSTRUCCIONES:

* + Prohibido consumo de alimentos
  + Prohibido equipo de diversión, celulares etc.
  + Prohibido jugar
  + Prohibido mover o intercambiar los equipos de los bancos de trabajo
  + Prohibido sacar los equipos del laboratorio sin autorización.
  + Ubicar los equipos y accesorios en el lugar dispuesto por el responsable del laboratorio, luego de terminar las prácticas.
  + Uso adecuado de equipos

## ACTIVIDADES POR DESARROLLAR:

x <- **c**(1,2,-3,4)

**if**(**all**(x>0)){ **print**("All Postives")

} **else**{

**print**("Not all positives")

}

1. What will this conditional expression return?

[1] "Not all positives"

1. Which of the following expressions is always FALSE when at least one entry of a logical vector xis TRUE?
2. **all(x)**
3. any(x)
4. any(!x)
5. **all(!x)**
6. The function nchar tells you how many characters long a character vector is. Write a line of code that assigns to the object new\_names the state abbreviation when the state name is longer than 8 characters.

library(dslabs)

data("murders")

ifelse(nchar(murders$state)>8, murders$abb, murders$state)



1. Create a function sum\_n that for any given value, say n, computes the sum of the integers from 1 to n (inclusive). Use the function to determine the sum of integers from 1 to 5,000.

suma\_n <- function(x){

s <- sum(x)

s

}

n <-5000

suma\_n(1:n)

[1] 12502500

1. Create a function altman\_plot that takes two arguments, x and y, and plots the difference against the sum.

altmat\_plot <- function(x, y){

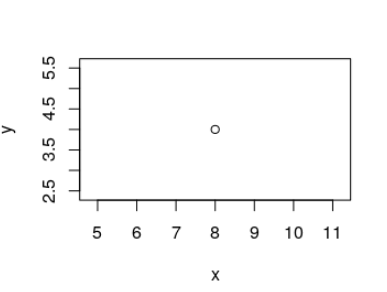
plot(x+y, x-y)

}

x <- 8

y <- 4

plot(x,y)



1. After running the code below, what is the value of x?

x<- 3

my\_func <- function(y){

x <- 5

y+5

}

x

x=3 fuera de la función; x= 5 dentro de la función

1. Write a function compute\_s\_n that for any give n computes the sum s n = 12 + 22 + 32 +42+. . . +n2. Report the value of the sum when n=10 .

compute\_s\_n <- function(n){

numero <- 1:n

sum(numero)

}

n=10

compute\_s\_n(n)

[1] 55

1. Define an empty numerical vector s\_n of size 25 using s\_n <- vector("numeric",

25) and store in the results of S1,S2,…S25 using a for-loop

s\_n <- vector("numeric", 25)

n <- 25

for(i in 1:n){s\_n[i] <- sum(i)

}

s\_n

[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

1. Repeat exercise 8, but this time use sapply.

s\_n <- vector("numeric", 25)

compute\_s\_n <- function(n){

sum(n)

}

n=1:25

s\_n <- sapply(n, compute\_s\_n)

s\_n

[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

1. Repeat exercise 8, but this time use map\_dbl.

install.packages("tidyverse")

install.packages("purrr")

library(purrr)

s\_n <- vector("numeric", 25)

compute\_s\_n <- function(n){

sum(n)

}

n <- 1:25

s\_n <- map\_dbl(n, compute\_s\_n)

s\_n

[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

1. Plot Sn versus n. Use points defined by n=1,…,25.

s\_n <- vector("numeric", 25)

compute\_s\_n <- function(n){

sum(n)

}

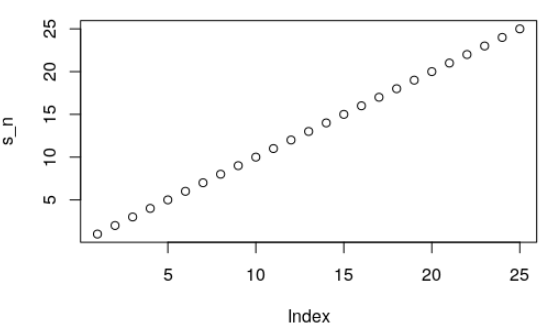
n=1:25

s\_n <- sapply(n, compute\_s\_n)

s\_n

plot(n)

plot(s\_n)



1. Confirm that the formula for this sum is Sn=n(n+1)(2n+1)/6.

s\_n <- vector("numeric", 25)

compute\_s\_n <- function(n){

sum(n)

}

n=1:25

compute\_s\_n(n)

s\_n <- sapply(n, compute\_s\_n)

s\_n

sn <-n\*(n+1)\*(2\*n+1)/6

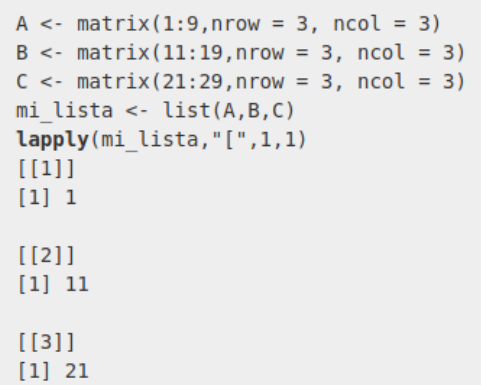
identical(s\_n, sn )

[1] FALSE

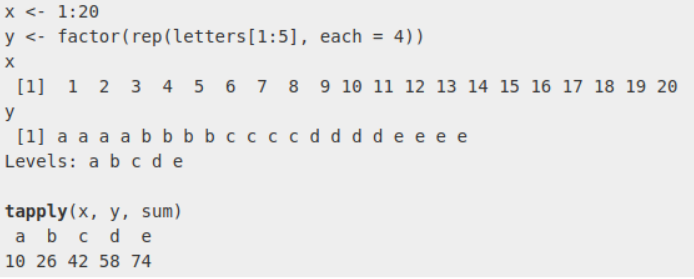
## INVESTIGACIÓN COMPLEMENTARIA (a elaborar por el estudiante)

Investigar acerca de las funciones lapply, tapply, mapply, vapply, y replicate

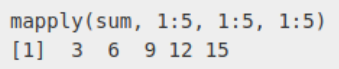
* **Laply**: Se diferencia con apply en que opera con listas. Recibe una lista y devuelve una lista



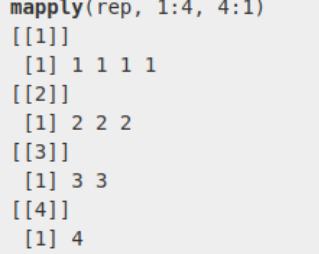
* **Tapply**: Realiza una operación (parámetro 3) respecto a un vector (parámetro 1) agrupada por los factores que se indiquen como argumento (parámetro 2).



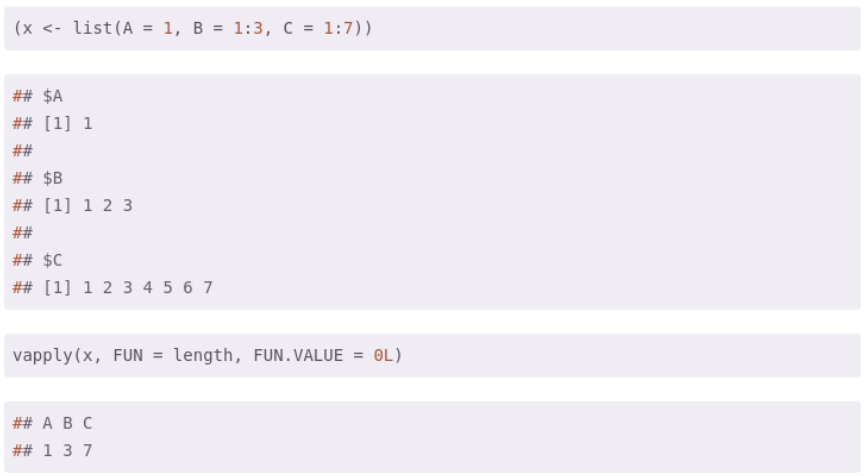
* **Mapply**: Realiza operaciones entre matrices y devuelve una lista o vector, a continuación se muestran algunas de las operaciones que admite: Suma el primer elemento de cada vector, después el segundo y así sucesivamente:



Repite cada elemento del primer vector el número de veces que indique el segundo vector:



* **Vapply**: Devuelve un vector con la longitud que tiene cada una de las listas introducidas como parámetro. Los bucles de la familia apply.



## DISCUSIÓN (a elaborar por el estudiante)

## los ejercicios planteados permiten asimilar el conocimiento de la utilización de R Estudios, dentro del cual se aprende a utilizar funciones, vectores, operaciones, bucles, condiciones etc.

1. **CONCLUSIONES (a elaborar por el estudiante)**

* R Studios dispone de muchas funciones que ahorra la elaboración de códigos tediosos y largos

1. **RECOMENDACIONES (elaborar por el estudiante)**

* Se recomienda practicar con diferentes ejercicios para asimilar lo conceptos nuevos que se va aprendiendo en R Studios

1. **BIBLIOGRAFÍA:**
   * A Brief History R: Past and Future History, Ross Ihaka, Statistics Department, The University of Auckland, Auckland, New Zealand, available from the CRAN website
   * What's new in R? What's new in R?

Firma del Presidente de Curso de Sexto A

Ing. Marlon Santiago Viñan Ludeña Mg. Sc DOCENTE CIS