INFORME SOBRE MODULO 2 TECNICAS Y HERRAMIENTAS

Módulo TyHM

JUAN PABLO FOURCADE

Universidad Nacional de Cuyo - Mendoza Argentina
pituf54@gmail.com

Gino Fragapane

Universidad Nacional de Cuyo - Mendoza Argentina
ginofragapane00@gmail.com

Tomas Juri

Universidad Nacional de Cuyo - Mendoza Argentina
tjuri02@gmail.com

Año 2024

Abstract

1 INTRODUCCION

En este informe se tratara sobre los temas tratados en el modulo 2

1.1 EJERCICIO 1 : PENITENCIA DE GAUSS

En este ejercicio mostramos como realizar una suma consecutiva de los elementos de un vector de manera automatica mediante la penitencia de gauss

```
# Sumar términos consecutivos de 1 a 1000
suma <- 0

for (i in 1:1000) {
  suma <- suma + i
}

# Mostrar el resultado
print(suma)
```

```
## [1] 500500
```

2 EJERCICIO 2 : ORDENAIENTO POR EL METODO DE LA BURBUJA

En este ejercicio debemos generar un vevtor y odenaro a traves de este metodo que consiste en lo siguiente:

La Ordenación de burbuja (Bubble Sort en inglés) es un sencillo algoritmo de ordenamiento. Funciona revisando cada elemento de la lista que va a ser ordenada con el siguiente, intercambiándolos de posición si están en el orden equivocado. Es necesario revisar varias veces toda la lista hasta que no se necesiten más intercambios, lo cual significa que la lista está ordenada. Este algoritmo obtiene su nombre de la forma con la que suben por la lista los elementos durante los intercambios, como si fueran pequeñas burbujas. También es conocido como el método del intercambio directo. Dado que solo usa comparaciones para operar elementos, se lo considera un algoritmo de comparación, siendo uno de los más sencillos de implementada.

```
# Tomo una muestra de 10 números ente 1 y 100
x<-sample(1:100,10)
# Creo una funci?n para ordenar
burbuja <- function(x){
n<-length(x)
for(j in 1:(n-1)){
for(i in 1:(n-j)){
if(x[i]>x[i+1]){
temp<-x[i]
x[i]<-x[i+1]
x[i+1]<-temp
}
}
}
return(x)
}
res<-burbuja(x)
#Muestra obtenida
x
```

```
##  [1] 60  9 61 90 68  7 23 59 12 83
```

3 EJERCICIO 3 : IMPLEMENTACION DE LA SERIE DE FIBBONACHI

La sucesión comienza con los números 0 y 1,2 a partir de estos, cada término es la suma de los dos anteriores, es la relación de recurrencia que la define. A los elementos de esta sucesión se les llama números de Fibonacci. Esta sucesión fue descrita en Europa por Leonardo de Pisa, matemático italiano del siglo XIII también conocido como Fibonacci. Tiene numerosas aplicaciones en ciencias de la computación, matemática y teoría de juegos. También aparece en configuraciones biológicas, como por ejemplo en las ramas de los árboles, en la disposición de las hojas en el tallo, en las flores de alcachofas y girasoles, en las incoerencias del brécol romanesco, en la configuración de las piñas de las coníferas, en la reproducción de los conejos y en como el ADN codifica el crecimiento de formas orgánicas complejas. De igual manera, se encuentra en la estructura espiral del caparazón de algunos moluscos, como el nautilus.

visto esto haremos una sucecion de fibbonachi que me sume hasta el numero 1.000.000 reaizando el siguiente codigo, el cual calculara cada termino hasta llegar al ultimo y dara las iteraciones realizadas.

```
# Definir la función de Fibonacci
fibonacci <- function(n) {
  if (n <= 1) {
    return(n)
  } else {
    return(fibonacci(n - 1) + fibonacci(n - 2))
  }
}

# Contar el número de iteraciones hasta 1.000.000
n <- 1
iteraciones <- 0
while (fibonacci(n) <= 1000000) {
  iteraciones <- iteraciones + 1
  n <- n + 1
```

```
}

# Mostrar el número de iteraciones
print(iteraciones)
```

```
## [1] 30
```

4 EJERCICIO 4 : MULTIPLICACION VECTORIAL

Ahora generaremos un codigo que me genere 2 vectores de 6 componentes, los multiplique y luego me tire como resultado el vector producto de ambos.

```
# Generar dos vectores de 6 componentes
vector1 <- c(1, 2, 3, 4, 5, 6)
vector2 <- c(7, 8, 9, 10, 11, 12)

# Calcular el producto vectorial
 producto <- (vector1 * vector2)

# Mostrar el vector producto
print(producto)
```

```
## [1]  7 16 27 40 55 72
```

5 EJERCICIO 5 : GRAFICOS

En este inciso explicaremos como se generan graficos en RMARKDOWN, explicado un codigo a continuacion.(ejemplo con edades de la provincia de mendoza)

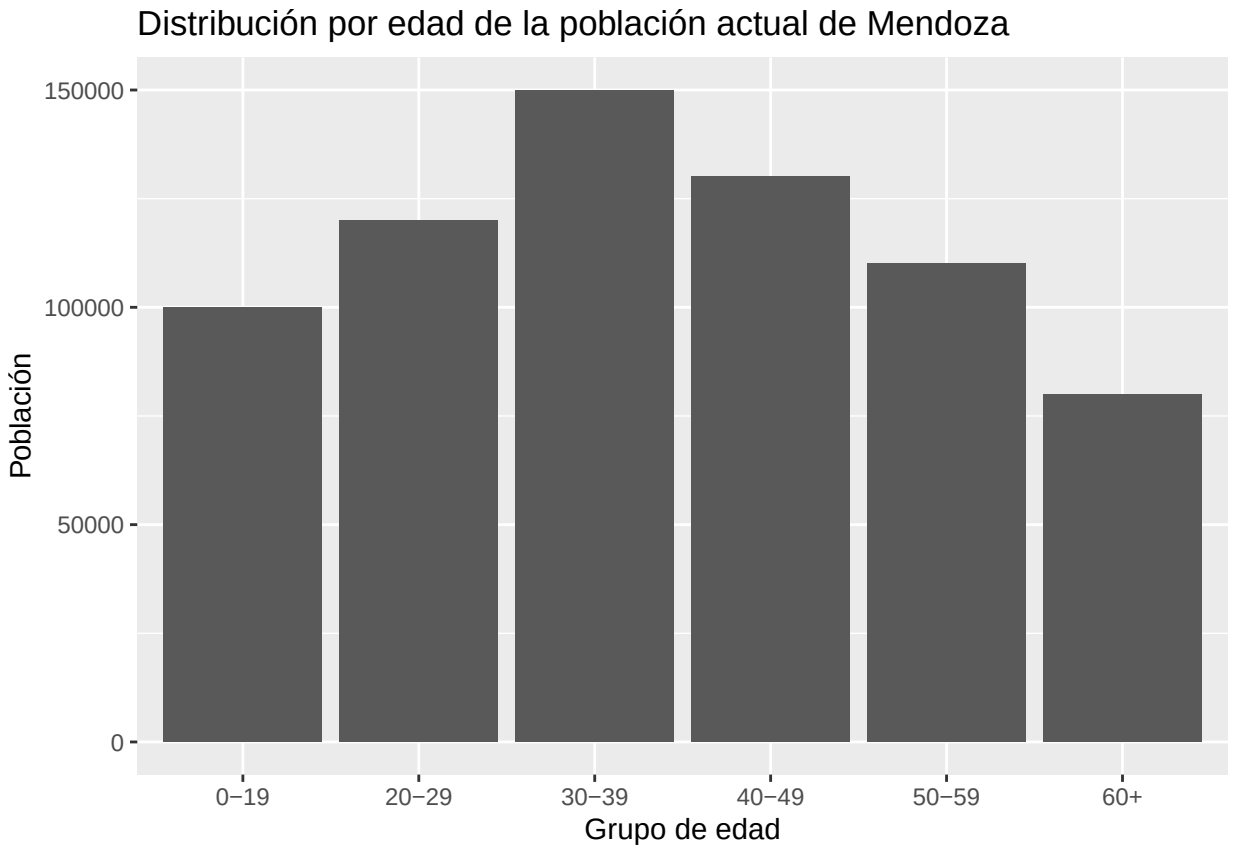
```
# Load libraries
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(ggplot2)

# Load data (replace with your actual data)
data <- data.frame(age_group = c("0-19", "20-29", "30-39", "40-49", "50-59", "60+"), population = c(100

# Create the graph
ggplot(data, aes(x = age_group, y = population)) +
  geom_bar(stat = "identity") +
  labs(title = "Distribución por edad de la población actual de Mendoza",
       x = "Grupo de edad", y = "Población")
```


Distribución por edad de la población actual de Mendoza
150000
100000
50000
0
Población
0−19
20−29
30−39
40−49
50−59
60+
Grupo de edad