

Bruno Vázquez Espinoza - A00838521

Cyrce Danae Salinas Rojas - A01666121

1. [3 pts] Crear un vector x con los datos 10, 7, 18, -11, 6, -34, 29

```
x <- c(10, 7, 18, -11, 6, -34, 29)
```

2. [12 pts] Calcula estadísticas simples del vector x. Calcular su media, desviación estándar y varianza. Crear un vector llamado estadx donde se guarden los 3 estadísticos.

```
estadx <- c(mean(x), sd(x), var(x))
```

3. [9 pts] Escribe un programa R para crear una secuencia de números del 10 al 80 y encuentra la media de los números del 25 al 100 y la suma de los números del 50 al 95.

```
seq <- 10:80  
mean_25_to_100 <- mean(seq[seq >= 25 & seq <= 100])  
sum_50_to_95 <- sum(seq[seq >= 50 & seq <= 95])
```

4. [7 pts] Escribe un programa R para crear un vector que contenga 10 valores enteros aleatorios entre -150 y +500. Tip: Revisa la función sample().

```
random_vector <- sample(-150:500, 10)
```

5. [12 pts] Escribe un programa R para obtener los primeros 10 números de Fibonacci. Puedes usar el siguiente código base usando un for para terminar tu código: fufb <- numeric(10) fb[1] <- fb[2] <- 1 Nota: Considera que los primeros 2 números de Fibonacci son 1 y 1.

```

fib <- numeric(10)
fib[1] <- 1
fib[2] <- 1

for (i in 3:10) {
  fib[i] <- fib[i-1] + fib[i-2]
}

fib

## [1] 1 1 2 3 5 8 13 21 34 55

```

6. [12 pts] Escribe un programa R para encontrar el valor máximo y mínimo de un vector dado sin utilizar las funciones base de `max()` y `min()`. Debes probar con los siguientes números: a. `c(-1, 20, -3, 40, -5, 60)` b. `c(10, 20, 30, -40, -50, -60)`

```

vector_a <- c(-1, 20, -3, 40, -5, 60)
vector_b <- c(10, 20, 30, -40, -50, -60)

max_value <- vector_a[1]
min_value <- vector_a[1]

for (i in 2:length(vector_a)) {
  if (vector_a[i] > max_value) {
    max_value <- vector_a[i]
  }
  if (vector_a[i] < min_value) {
    min_value <- vector_a[i]
  }
}

max_value

```

```
## [1] 60
```

```
min_value
```

```
## [1] -5
```

```

max_value <- vector_b[1]
min_value <- vector_b[1]

for (i in 2:length(vector_b)) {
  if (vector_b[i] > max_value) {
    max_value <- vector_b[i]
  }
}

```

```

    if (vector_b[i] < min_value) {
      min_value <- vector_b[i]
    }
  }

max_value

```

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

```
min_value
```

```
## [1] -60
```

7. [15 pts] Escribe una función de R llamada “multiplica” para multiplicar dos vectores de tipo entero y longitud n, de la misma longitud ambos. Probar con los siguientes valores: `multiplica(c(30, 10), c(3,4))` # Salida: `[1] 90 40`

```

multiplica <- function(vector1, vector2) {
  if (length(vector1) != length(vector2)) {
    stop("Los vectores deben tener la misma longitud.")
  }

  resultado <- vector1 * vector2
  return(resultado)
}

multiplica(c(30, 10), c(3, 4))

```

```
## [1] 90 40
```

8. [15 pts] Escribe una función de R llamada “cuenta” para contar cuántas veces aparece un valor específico en un vector dado. Probar con los siguientes valores: `cuenta(c(10, 2, 10, 7, 2, 7, 2),2)` # Salida: `[1] 3`

```

cuenta <- function(vector, valor) {
  contador <- 0

  for (i in 1:length(vector)) {
    if (vector[i] == valor) {
      contador <- contador + 1
    }
  }
}

```

```

    }

    return(contador)
}

cuenta(c(10, 2, 10, 7, 2, 7, 2), 2)

```

```
## [1] 3
```

9. [15 pts] Escribe una función de R llamada “enésimo” para extraer cada enésimo elemento de un vector dado. Probar con los siguientes valores: enesimo(1:100, 5) # Salida: [1] 1 6 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86 91 96

```

enesimo <- function(vector, n) {
  resultado <- vector[seq(vector[1], length(vector), n)]
  return(resultado)
}

enesimo(1:100, 5)

```

```
## [1] 1 6 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86 91 96
```