2.5 Ejercicios propuestos

2.5.1 Ejercicios de vectores

1. Genera una secuencia de longitud 100 con los primeros valores múltiplos de 4 y súmalos. Usa la función `seq()` y `sum()`.
2. Usando la función ' rep()` genera un vector de tamaño 1000 con los 100 1ºs valores por 0, los 100 siguientes por 1, … y los 100 últimos por 9.
3. Usando la función `sample()` genera 100 valores aleatorios con reemplazamiento entre 1 y 1000. Muestra por pantalla los índices y valores (ordenados y no) donde los valores son múltiplos de 3. Puede ser de utilidad la función `which()`.
4. Dados los vectores `x`, `y`, `z` dados por: de 1 a 10, de 26 a 35 y de 100 a 91, realiza las siguientes operaciones:
   a. Suma de los tres vectores.
   b. Producto de los tres vectores.
   c. Producto escalar de `x` e `z`. Resuelve usando y sin usar la función `%*%`.
5. Simula 500 tiradas de una moneda con resultados equiprobables *cara* y *cruz*. ¿Cuantas caras y cruces han salido? ¿Cuál es el porcentaje del resultado *cara* en las 500 tiradas? Hint: Funciones `sample()`, `sum()`, `mean()`

2.5.2 Ejercicios de matrices y arrays

1. Genera una matriz 100×10 con valores columnas de 0 a 9.
2. Sea el experimento, tirar aleatoriamente 100 valores de un dado equibrado.
   a. Obtén una realización del experimento.
   b. Obten dos realizaciones y *pega* los dos vectores creando una matriz 100×2.
   c. Repite el proceso (vectores distintos) y obtén una matriz 100×5.
   d. Asigna nombre de columnas *tirada-1*, …, *tirada-5* (usa la función `paste()`).
   e. Obtén la frecuencia relativa por columnas de la matriz del suceso *número <3* (usa la función `colSums()`.
3. Con la matriz del ejercicio anterior:
   a. ¿Cuántas y cuáles filas salío el la segunda columna el valor 6?
   b. Genera la submatriz formada por las filas de la 20 a la 25 incluidas y la 2 y cuarta columna.
   c. Elimina la 3 columna y las 10 últimas filas.
   d. Obtén la matriz que verifique e la 1ª columna es mayor que 4.
4. Dado los puntos del espacio $A^t = (1, 1, 1), B^t = (1, 2, 3), C^t = (-1, 2, -1)$ calcula la distancia de esos puntos al punto $O^t < -(-1, -1, -1)$. Para ello genera una matriz con los puntos A, B y C y obten la distancia euclídea. Usa la función `rbind()` y `rowSums()`. Comprueba que el resultado coincide con la función de R `dist()`.

2.5.3 Ejercicios de data.frames y listas

1. La ejecución `data(iris)` trae a memoria el `data.frame` *iris*. Consulta la ayuda sobre este conjunto de datos.
   a. Comprueba que efectivamente es un `data.frame`. Usa las funciones `class()` y/o `str()`. Convierte el conjunto en una matriz (usa la función `as.matrix()`. ¿Qué diferencias se observan? Describe el `data.frame` usnado la función `summary()`. Calcula cuántas flores de cada especie contiene el data frame iris (puedes usar la función `table()`).
   b. Genera los `data.frame` correspondiente a cada subespecie de iris. Usa la función `levels()` del factor `Species`.
   c. Visualiza como matriz y lista el elemento fila =5 para la variable `Petal.Width`.
   d. Obtén el promedio de la varible `Petal.Length` para la especie *versicolor* con *Sepal.Length>6.3*. Usa la función `mean()` para obtener el promedio de un vector numérico
   e. Transforma la variable `Petal.Length` a escala logarítmica.
   f. Usando la función `colMeans()` obtén el valor promedio de las longitudes y ancho del pétalo y sépalo.
   g. Ordena el `data.frame` en sentido creciente en función de la variable `Petal.Length`. Visualiza la cabecera del `data.frame` (usa la función `head()`).

h. Genera un `data.frame` que sea una muestra aleatoria de 10 flores. Usa la función `sample()`.

2. Dada la lista `l <- list( Species = sample(iris$Species,size=20), iris=iris, m = matrix(1:100, ncol=4))`

   a. Haz un resumen de cada componente de la lista. Usa la función `summary()`. Puede ser de utilidad la función `lapply()`.

   b. Añade un nuevo elemento de la lista que sea el valor "Hello world".

   c. Genera una nueva lista de nombre *ll* que sean dos elementos al azar de los 4 dados.

   d. Determina el último elemento de la segunda componente de la lista generada en el apartado anterior. Hazlo a partir de `ll[[2]]` y `ll[2]`. Justifica que debe salir en cada caso.

2.5.4 Ejercicios de funciones

1. Crea una función que devuelva el "Hello wold" para los idiomas es, en y gl. Como argumento el código de idioma indicado.

```
Fun.Hello(idioma="en")
```

```
## [1] "Hello world"
```

```
Fun.Hello(idioma="es")
```

```
## [1] "Hola mundo"
```

```
Fun.Hello(idioma="gl")
```

```
## [1] "Ola mundo"
```

2- Construye una función para que resuelva una ecuación de 2º grado dado las forma $ax^2 + bx + c = 0$ dados *a, b, c* valores que aseguran la solución.

```
a. Comprueba que para $x^2-3x+2$ la solución es `(2,1)`.

b. Rehaz la función para que permita los argumentos a, b y c como vectores. No uses bucles, apoyat
$x^2-3x+2$ y $x^2-4x+4$ las soluciones son `(2,1)` y `(-2,-2)`.
```

```
sol(a=1,b=-3, c=2)
```

```
##      x1 x2
## [1,]  2  1
```

```
sol(b=c(-3,4), c=c(2,4))
```

```
##      x1 x2
## [1,]  2  1
## [2,] -2 -2
```

3. Construye una función que calcule la frecuencia relativa del suceso cara en el lanzamiento de una moneda con probabilidad de cara *p* *n* veces (por defecto *p=0.5* y *n=100*).

   a. Ejecutala para n=100, 1000, 5000.

   b. Programa la función para que permita que el argumento n sea un vector. Usa la función `Vectorize()`.

   c. Genera un gráfico donde el eje de las X sea la secuencia de valores para n de 100, 1000, 2000, 3000, …, 10000 y en el eje de las Y la frecuencia relativa de aparición del resultado *cara*. Además dibuja la línea horizontal (de color azul) en el valor teórico de *p*. Usa las funciones `plot()` y `abline()`.

```
#a.
lanzamiento(n=100)
```

```
## [1] 0.48
```

```
lanzamiento(n=1000)
```

```
## [1] 0.48
```

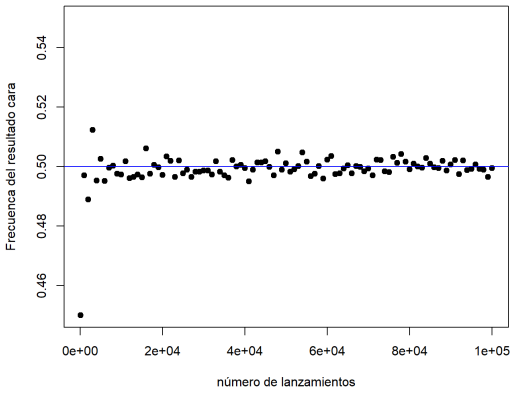
```
lanzamiento(n=5000)
```

```
## [1] 0.503
```

```
#b.
v.lanzamiento(n=c(100,1000,5000))
```

```
## [1] 0.4800 0.4910 0.5022
```

```
#c.
n <- c(100, seq(1000,100000,1000))
```



2.5.5 Ejercicios de Estructura condicionales y bucles

1. Dada un valor redondealo al entero más proximo. Usa la función `trunc()` y después la instrucción `if`
2. Repite el ejercicio anterior usando `ifelse` .
3. Calcula la suma de los 10 primeros valores de la sucesión dada por

$$a_1 = 1, a_n = \begin{cases} log(a_{n-1} + 2) & \text{si } n \text{ es par} \\ exp(a_{n-1} + 2) & \text{si } n \text{ es impar} \end{cases}$$

```
## [1] 11434.58
```

2.5.6 Ejercicios de importación y exportación de datos

Archivos en el repositorio de datos https://archive-beta.ics.uci.edu/

1. El conjunto de datos *wine* contiene datos de análisis químico de vinos para determinar su origen. Concretamente en el archivo *wine.data* están los datos y en el archivo *wine.names* aparecen los nombres de las variables/columnas (salvo la 1ª que es *class*). Genera un objeto de la clase `data.frame()` con estos datos. A continuación se muestran las 3 primeras y las 2 últimas columnas del `data.frame()` :

class	Alcohol	Malic acid	OD280/OD315 of diluted wines	Proline
1	14.23	1.71	3.92	1065
1	13.20	1.78	3.40	1050
1	13.16	2.36	3.17	1185
1	14.37	1.95	3.45	1480
1	13.24	2.59	2.93	735
1	14.20	1.76	2.85	1450

2. El conjunto de datos *Spambase* contiene datos de 4601 correos electrónicos clasificados en *spam/no spam*. Concretamente 57 variables con información del correo-e y la variable 58 es binaria indicando si el correo-e es spam o no.
    a. Importa el conjunto de datos a un objeto `data.frame()` .

b. Usando `read.table()` abre el archivo de texto que tiene los nombres de las variables. Usa los argumentos `skip` , `nrows` y `comment.char` . Ten en cuenta que la última variable (columna 58) es la indicadora de la clase del correo-e *spam/no spam*, asígnale el nombre *class*.

c. Transforma la columna *class* en `factor()` con los niveles *spam/no_spam*. ¿Qué porcentaje de correos-e *spam* hay en el archivo?

A continuación se muestran las 3 primeras y las 2 últimas columnas del archivo

word_freq_make	word_freq_address	word_freq_all	capital_run_length_total	class
0.00	0.64	0.64	278	1
0.21	0.28	0.50	1028	1
0.06	0.00	0.71	2259	1
0.00	0.00	0.00	191	1
0.00	0.00	0.00	191	1
0.00	0.00	0.00	54	1