# Tarea 11 - Dataframes con el módulo `pandas`

Curso de `Python`

## Ejercicio 1

Crea un dataframe con 15 filas y 2 columnas. La primera columna se llamará `x`, la segunda `y`. Cada entrada será un número real a tu elección. Guarda el dataframe en una variable llamada `points`.

### Solución

```
import pandas as pd
data = {"x": [0, 0.25, -0.5, -0.75, 1, 1.25, -1.5, -1.75,
              2, 2.25, -2.5, -2.75, 3, 3.25, -3.5],
        "y": [0, 0.25, 0.5, -0.75, -1, 1.25, 1.5, -1.75,
              -2, 2.25, 2.5, -2.75, -3, 3.25, 3.5]}
points = pd.DataFrame(data = data)
```

## Ejercicio 2

Del dataset `points`, muestra las filas cuyo valor en la columna `x` sea positivo.

### Solución

```
points[points["x"] > 0]
```

## Ejercicio 3

Del dataset `points`, muestra las filas cuyo valor en la columna `y` sea negativo. Usa el método `.query()`.

### Solución

```
points.query("y < 0")
```

## Ejercicio 4

Del dataset `points`, muestra las observaciones cuyos puntos (`x`, `y`) pertenezcan al primer cuadrante. Usa el método `.query()`.

**Solución**

```
points.query("x >= 0 and y >= 0")
```

## Ejercicio 5

Del dataset `points`, muestra las observaciones de la forma: "El punto ({x}, {y}) {no/sí} pertenece al primer cuadrante".

**Solución**

```
for p in points.itertuples():
  print("El punto ({}, {}) {} pertenece al primer cuadrante".
        format(p[1], p[2], "" if p[1] >= 0 and p[2] >= 0 else "no"))
```

## Ejercicio 6

Crea un dataframe con 10 filas y 4 columnas. La primera columna se llamará `word`; la segunda, `length`; la tercera, `vowels`; y la última, `consonants`. La columna `words` contendrá las siguientes 10 palabras: "euro", "diez" "algas", "broma", "cicuta", "fatiga", "nachos", "jadeos", "hazañas", "boutique". Las columnas `length`, `vowels` y `consonants` contendrán, respectivamente, la longitud, el total de vocales y el total de consonantes. Guarda el dataframe en la variable `words`.

**Solución**

```
def total_vowels_and_consonants(w):
  """
  Devuelve el total de vocales de una palabra
  Args:
    w: Palabra en formato string
  Returns:
    (vowels, consonants): Tupla de números enteros
  """
  vowels = 0
  consonants = 0
  for c in w:
    if c in ["a", "e", "i", "o", "u"]:
      vowels += 1
    else:
      consonants += 1
  return (vowels, consonants)

import pandas as pd
words = ["euro", "diez", "algas", "broma", "cicuta",
         "fatiga", "nachos", "jadeos", "hazañas", "boutique"]

data = {"word": words,
```

```
        "length": map(len, words),
        "vowels": list(map(lambda w: total_vowels_and_consonants(w)[0], words)),
        "consonants": list(map(lambda w: total_vowels_and_consonants(w)[1], words))}

words = pd.DataFrame(data = data)
```

## Ejercicio 7

Muestra las 10 observaciones de `words` con el formato "La palabra {word} tiene {length} letras, de las cuales {vowels} son vocales y {consonants} consonantes".

### Solución

```
for row in words.itertuples():
  print("La palabra {} tiene {} letras, de las cuales {} son vocales y {} consonantes".
        format(row[1], row[2], row[3], row[4]))
```

## Ejercicio 8

Muestra aquellas observaciones de `words` que tienen el mismo número de vocales y consonantes. Usa el método `.query()`.

### Solución

```
words.query("vowels == consonants")
```

## Ejercicio 9

Investiga el método `.sort_values()` para ordenar las observaciones según la longitud de las palabras en orden descendente.

### Solución

```
words.sort_values(by = "length", ascending = False)
```

## Ejercicio 10

Convierte la columna `vowels` a lista y, con `sorted()` ordénala de mayor a menor. Investiga para ello el método `.tolist()`.

### Solución

```
sorted(words["vowels"].tolist(), reverse = True)
```