1. **Contexto. Por qué el sitio web elegido proporciona dicha información.**

En enero de 2020, el famoso Covid-19 llegó a España forzando la decisión de confinamiento para todas las comunidades desde mediados de marzo, cuando el número de contagios empezó a aumentar drásticamente. Debido a esta situación, la mayoría de las empresas se han visto afectadas por la imposibilidad de abrir sus oficinas, tiendas, o establecimientos en general dado que los empleados deben mantenerse en casa. En este sentido, dado que muchas empresas no pueden ofrecer sus servicios/productos (muchos de ellos presenciales), la economía ha recibido un importante impacto.

Este impacto se distribuye a lo largo de un continuo en el que influye directamente la capacidad de las empresas para continuar produciendo/ofreciendo/distribuyendo sus servicios/productos a distancia. Por ejemplo, si existe una empresa que a pesar de mantener a sus empleados teletrabajando puede seguir produciendo, aunque la venta online les sea más difícil, pueden resistir mejor que otra empresa que tiene que dejar de producir porque no tienen un sitio físico en el que generar el producto y, por tanto, no pueden venderlo.

Si la empresa cotiza en bolsa, una manera de observar mejor el impacto de esta pandemia es estudiar las variaciones en el precio de las acciones. De esta forma, se pueden analizar (dado que ya ha pasado unas cuantas semanas desde el inicio del confinamiento), los cambios que se han producido, por ejemplo, con respecto al inicio del año o las mismas épocas de años anteriores.

En este sentido, el código extrae un dataset con información (en euros) sobre las cotizaciones en bolsa (apertura, máximo, mínimo, cierre y cierre ajustado) de la empresa Telefónica, desde el 2015-04-16 hasta la fecha actual 2020-04-14, utilizando, principalmente las librerías: selenium, beautifulsoup y pandas junto con numpy.

El sitio web elegido es Yahoo Finance por su confiabilidad y por facilitar el campo cierre ajustado, que otras webs de datos financieros no ofrece. Además, este sitio web lleva muchos años dando servicios de información y noticias financieras actualizadas, lo que lo convierte en una fuente de datos segura y fiable.

2. **Definir un título para el dataset**

EVOLUCIÓN DE LAS COTIZACIONES DE TELEFÓNICA DURANTE LOS ÚLTIMOS 5 AÑOS

3. **Descripción del dataset.**

El dataset tiene como objetivo principal servir de base para comprender cuál, en términos de cantidad y dinero, ha sido el impacto que ha tenido la pandemia, concretamente en la empresa Telefónica.

Es por ello por lo que contiene datos de cotizaciones de la empresa desde el 16 de abril de 2015 hasta el 14 de abril de 2020 (fecha actual del proyecto). El periodo abarca un periodo temporal de aproximadamente 5 años. Hay que tener en cuenta, sin embargo, que el número de registros (días) del dataset no se corresponde con 1.825 (el correspondiente a aproximadamente 5 años), dado que la bolsa no está activa durante los fines de semana y algunos festivos.

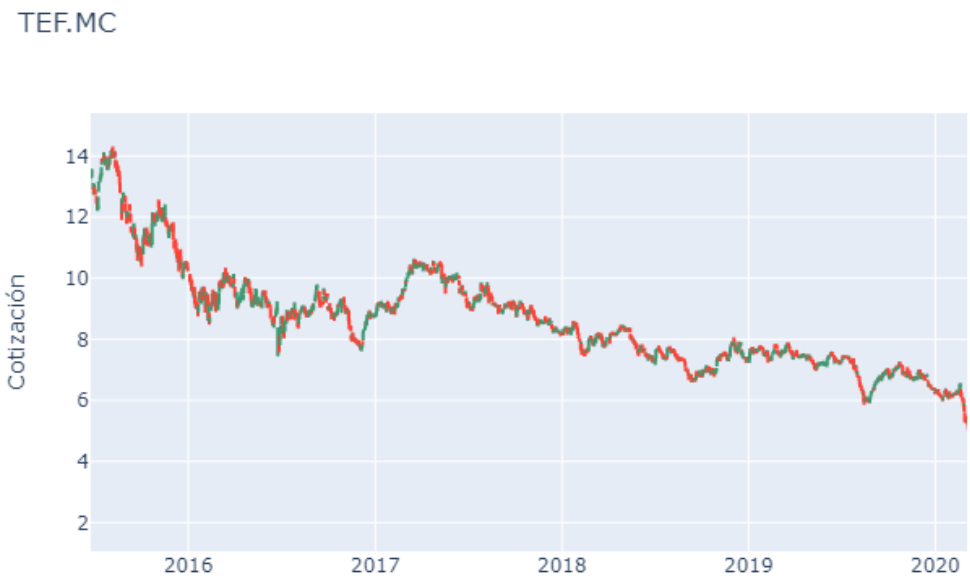
Como ventajas del programa en general, cabe destacar la flexibilidad que aporta a la extracción de información ya que permite consultar diferentes periodos de tiempo y diferentes empresas de interés. Además, a pesar de que necesita la descarga de un componente externo

Autores:
- Enrique Javier Andrés Orera
- Rubén Silva Marín

(driver para la librería Selenium) este script lo hace de forma automática y posteriormente lo borra para que no haya archivos innecesarios en el ordenador de la persona que lo ejecuta. Como ventajas adicionales, cabe incluir la aleatorización de user-agents y la posibilidad de agregar retardos en las peticiones de Selenium. Finalmente, el script que extrae los datos, lo hace también teniendo en cuenta algunos puntos como el reparto del dividendo (que se refleja como un registro diferente al resto con solo 2 datos) el cual se elimina, o la posibilidad de que la página haya perdido datos (como el del 25 de diciembre de 2019 para telefónica) que aparecerían con un guion " - ", y, también se eliminan en el script.

Por otro lado, el conjunto de datos extraídos sí necesita limpieza solamente en el caso de telefónica, dado que el registro del 25 de diciembre de 2019 parece ser erróneo, aunque se ha dejado por si se quiere tener en cuenta en los análisis. También es cierto que la información que ofrece está limitada a estudiar un objetivo en particular (impacto en las cotizaciones de una empresa), y ofrece exclusivamente los campos descritos más abajo, sin posibilidad de ampliación. No limita, sin embargo, el análisis de otras cuestiones siempre que se puedan llevar a cabo a través de la información proporcionada en este dataset.

4. **Representación gráfica. Presentar una imagen o esquema que identifique el dataset visualmente**



5. **Contenido. Explicar los campos que incluye el dataset, el periodo de tiempo de los datos y cómo se ha recogido.**

El dataset está compuesto por 1280 registros (filas) con los siguientes campos (columnas):

- Date: dato de tipo fecha que indica el día en la que fueron recogidos los demás datos.

- High: dato de tipo numérico que indica la cotización máxima que alcanzaron las acciones de Telefónica en la fecha indicada.

- Low: dato de tipo numérico que indica la cotización mínima que alcanzaron las acciones de Telefónica en la fecha indicada.

Autores:
- Enrique Javier Andrés Orera
- Rubén Silva Marín

- Open: dato de tipo numérico que indica la cotización de apertura (inicial) de las acciones de Telefónica en la fecha indicada.

- Close: dato de tipo numérico que indica la cotización de cierre (final) de las acciones de Telefónica en la fecha indicada.

- Volume: dato de tipo numérico que indica el volumen de operaciones que se llevaron a cabo con las acciones de Telefónica en la fecha indicada.

- Adj Close: dato de tipo numérico que indica la cotización final de las acciones de Telefónica en la fecha indicada. El cierre ajustado es el precio de cierre después de los ajustes para todas las distribuciones de splits y dividendos aplicables. Los datos se ajustan utilizando los multiplicadores de splits y dividendos correspondientes, de conformidad con las Center for Research in Security Prices (CRSP).

(Fuente: https://es.ayuda.yahoo.com/kb/%C2%BFQu%C3%A9-es-el-cierre-ajustado-sln28256.html visitada 29/03/2020)

El periodo recogido incluye aproximadamente los 5 años anteriores a la fecha actual: desde el 30 de marzo de 2015 hasta el 28 de marzo de 2020.

Para la extracción del dataset, se ha optado por las librerías:

- **Argparse** para pasar los argumentos de *Empresa*, *Fecha inicial*, y *Fecha final* al programa. Por defecto estos son "TEF.MC", 5 años antes de la fecha del día en el que se ejecuta el script, y la fecha del día en el que se está ejecutando el script, respectivamente;
- **Random** para asignar aleatoriamente un *user-agent* a la petición de la librería **Selenium**;
- **Requests**, **os**, **platform** y **zipfile** para descargar el archivo driver necesario para **Selenium**, en función del sistema operativo en el que se está ejecutando el script (solo acepta Windows y Linux de momento);
- **Selenium** para realizar una consulta de manera más avanzada dado que con la librería **Requests** no era posible obtener toda la información (la página cargaba más información a medida que se iba haciendo scroll-down hacia el final);
- **Beautifulsoup** para traducir el documento html descargado y buscar los elementos que interesan para el análisis;
- **Pandas** y **Numpy** para manejar los datos volcados con un formato de dataframe y poder volcarlos en un archivo CSV; y
- **Plotly**, concretamente el módulo de **graph_objects** para realizar el gráfico temporal de los datos y volcarlos en un archivo HTML.

El proceso de extracción se hace apoyándose en la etiqueta *td* del documento HTML ya que son los datos individuales de la tabla de cotizaciones. Cada *td* se va uniendo individualmente a una lista llamada datos que luego se reestructura para que tenga las 7 columnas requeridas.

6. Agradecimientos

Los datos han sido recogidos de la base de datos online de Yahoo Finance. Almacena bases de datos históricas de cotizaciones (ofreciendo además el cambio de cierre ajustado, un hecho diferencial frente a otras fuentes) . Es un sitio web confiable que lleva presentando información financiera a los usuarios de su página web muchos años.

Autores:
- Enrique Javier Andrés Orera
- Rubén Silva Marín

Utilizando para recoger los datos el lenguaje de programación Python y la librerías reseñadas en el quinto apartado de este documento. Se desconoce si existen análisis anteriores de estos mismos datos.

7. Inspiración

Dada la situación actual en la que se ve sumergida el mundo entero por el COVID-19, se ha dispuesto el siguiente dataset para evaluar el impacto económico que ha tenido en la empresa Telefónica. Esto se ha llevado a cabo seleccionando el periodo de los 5 años anteriores.

Este dataset puede ser utilizado para ver si las cotizaciones en este periodo de pandemia son significativamente diferentes (inferiores, por lo general) a las que había en la misma época de años anteriores o directamente en periodos anteriores.

También puede ser utilizado para investigar la posible existencia de patrones estacionales de cotizaciones.

Se desconoce si existen análisis anteriores de estos mismos datos, es posible que sí, pero seguramente enfocados desde un punto de vista crematístico, no de investigación.

8. Licencia. Seleccione una de estas licencias para su dataset y explique el motivo de su selección:

Se han de tener en cuenta las siguientes consideraciones:

- La información extraída de la página de Yahoo Finance u otras entidades asociadas, no puede utilizarse con fines comerciales por ninguno de sus usuarios y se establecen restricciones a su difusión.
- Zenodo, la página de destino para el dataset de esta práctica está certificada por OPENAire, lo que lo convierte en Open Data Source.
- Teniendo esto en cuenta, la publicación del dataset en Zenodo no se establecería como una actividad comercial.
- En base a lo anterior se establece que la posible licencia para aplicar es: Other(Not Open) .

9. Código. Adjuntar el código con el que se ha generado el dataset, preferiblemente en Python o, alternativamente, en R.

```
import requests as rq

from zipfile import ZipFile

from selenium import webdriver

import os

from bs4 import BeautifulSoup as bsoup

import numpy as np

import pandas as pd

import plotly.graph_objects as go

import datetime
```

Autores:
- Enrique Javier Andrés Orera
- Rubén Silva Marín

```
import argparse
import time
import random
import platform

# Analizamos los argumentos de la linea de comandos, incluimos valores por defecto

parser = argparse.ArgumentParser()
parser.add_argument("--ticker", help="Enter corporation ticker symbol", default='TEF.MC')
parser.add_argument("--startDate", help="Enter start date of interval YY-MM-DD",
                    default=str(datetime.date.today() - datetime.timedelta(days=1825)))
parser.add_argument("--endDate", help="Enter end date of interval YY-MM-DD", default=str(datetime.date.today()))
args = parser.parse_args()

# Se asignan los valores de entrada requeridos

ticker = args.ticker
startDate = datetime.datetime.strptime(str(args.startDate), "%Y-%m-%d")
endDate = datetime.datetime.strptime(str(args.endDate), "%Y-%m-%d")

# Se transforman las variables de fechas para que se puedan utilizar en la url, ya que Yahoo aplica los filtros mediante este formato de Unix

period1 = int(startDate.timestamp())
period2 = int(endDate.timestamp())

# Configuración de la url que contiene los datos para que se filtre por empresa (ticker) y fechas deseadas (startDate y endDate)

url = "https://finance.yahoo.com/quote/%s/history?period1=%s&period2=%s&interval=1d" % (ticker, period1, period2)

# Lista de user agents posibles para ir rotando a la hora de realizar las consultas

userAgents = [
    # Chrome
    'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.113 Safari/537.36',
    'Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.90 Safari/537.36',
    'Mozilla/5.0 (Windows NT 5.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.90 Safari/537.36',
    'Mozilla/5.0 (Windows NT 6.2; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.90 Safari/537.36',
```

Autores:

- Enrique Javier Andrés Orera
- Rubén Silva Marín

```
    'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/44.0.2403.157 Safari/537.36',
    'Mozilla/5.0 (Windows NT 6.3; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.113 Safari/537.36',
    'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/57.0.2987.133 Safari/537.36',
    'Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/57.0.2987.133 Safari/537.36',
    'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/55.0.2883.87 Safari/537.36',
    'Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/55.0.2883.87 Safari/537.36',
    # Firefox
    'Mozilla/4.0 (compatible; MSIE 9.0; Windows NT 6.1)',
    'Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko',
    'Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0)',
    'Mozilla/5.0 (Windows NT 6.1; Trident/7.0; rv:11.0) like Gecko',
    'Mozilla/5.0 (Windows NT 6.2; WOW64; Trident/7.0; rv:11.0) like Gecko',
    'Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko',
    'Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.0; Trident/5.0)',
    'Mozilla/5.0 (Windows NT 6.3; WOW64; Trident/7.0; rv:11.0) like Gecko',
    'Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)',
    'Mozilla/5.0 (Windows NT 6.1; Win64; x64; Trident/7.0; rv:11.0) like Gecko',
    'Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.1; WOW64; Trident/6.0)',
    'Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.1; Trident/6.0)',
    'Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0; .NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR
3.5.30729)'
]


# Seleccionamos user agent aleatoriamente


userAgent = random.choice(userAgents)


# Se descarga el driver necesario para trabajar con Selenium y Google Chrome. El if-statement sirve para discriminar el driver que
se debe descargar en función del sistema operativo


print(platform.system())
print(platform.release())


if platform.system() == "Windows":
    driver = rq.get('http://chromedriver.storage.googleapis.com/80.0.3987.106/chromedriver_win32.zip')


    with open('chrome_driver.zip', 'wb') as d:
        d.write(driver.content)
```

Autores:

- Enrique Javier Andrés Orera
- Rubén Silva Marín

```
    with ZipFile('chrome_driver.zip', 'r') as zip:
        zip.extractall()

elif platform.system() == "Linux":
    os.system('sudo apt-get install chromium-browser')
    driver = rq.get('http://chromedriver.storage.googleapis.com/80.0.3987.106/chromedriver_linux64.zip')

    with open('chromedriver_linux64.zip', 'wb') as d:
        d.write(driver.content)

    with ZipFile('chromedriver_linux64.zip', 'r') as zip:
        zip.extractall()
    print("descomprimido")
    os.system('chmod +x chromedriver')
    os.system('sudo mv -f chromedriver /usr/local/share/chromedriver')
    os.system('sudo ln -s /usr/local/share/chromedriver /usr/local/bin/chromedriver')
    os.system('sudo ln -s /usr/local/share/chromedriver /usr/bin/chromedriver')

# Se eligen las opciones que se requieren para el driver de Selenium. Headless sirve para ocultar el driver y que no se vea mientras navega por la web.
# Los 2 siguientes sirven para evitar mensajes extra en la ejecución y el último es para establecer el user-agent

options = webdriver.ChromeOptions()
options.add_argument('headless')
options.add_argument('--ignore-certificate-errors')
options.add_argument('--ignore-ssl-errors')
options.add_argument('user-agent=%s' % (userAgent))

# Creamos objeto webdriver (selenium), que es el que realiza la petición con las opciones anteriormente establecidas

driver = webdriver.Chrome(options=options)

# Se aplica el retardo en la petición de 1 segundo. Se puede configurar para mayor o menor tiempo
seconds = 1
driver.implicitly_wait(seconds)

# Se abre una nueva petición a la página deseada
```

Autores:

- Enrique Javier Andrés Orera
- Rubén Silva Marín

```
driver.get(url)


# Ahora el driver de selenium se configura para hacer click en el botón de aceptar de un mensaje de información que aparece


agent = driver.execute_script("return navigator.userAgent")

print(agent)


driver.find_element_by_xpath('//button[text()="Acepto"]').click()


# Se establecen 2 variables para comparar los archivos HTMLs de la página que se usarán en el bucle


html1 = driver.page_source

html2 = 0


# Se configura el final de la página al que se quiere llegar


end = driver.find_element_by_xpath('//span[contains(text(), "*Close price adjusted for splits.")]')


# Leemos la página hasta llegar al final de la tabla (ScrollDown) indefinidamente mientras los archivos HTMLs no coincidan.

# Esto se ha hecho así porque siempre que se pueda hacer ScrollDown, el HTML de la página cambiará con respecto al antrior.

# Así, el bucle termina cuando los documentos HTML anterior al ScrollDown y posterior coincidan, de forman que se habría llegado al final de la página


while html1 != html2:

   html1 = driver.page_source

   time.sleep(2)

   driver.execute_script('arguments[0].scrollIntoView(true);', end)

   html2 = driver.page_source


# Se asigna el documento HTML traducido por Beautifulsoup a la variable que se requiere y se cierra el driver de Selenium.

# Se aplica un retardo entre acciones para que no haya problemas para guardar los datos, ya que si el driver se cierra antes de que se termine de extraer los datos,

# probablemente dará problemas


soup = bsoup(html2, features='lxml')


driver.implicitly_wait(seconds)


driver.close()
```

Autores:

- Enrique Javier Andrés Orera
- Rubén Silva Marín

```
# Una vez que el driver se ha cerrado, se procede a eliminar los archivos del driver según el sistema operativo en el que se esté ejecutando


if platform.system() == "Windows":
    os.system('TASKKILL /F /IM chromedriver.exe')


if platform.system() == "Windows":
    time.sleep(2)
    os.remove('chromedriver.exe')
    os.remove('chrome_driver.zip')


# Se crea una lista vacía y mediante un bucle for, se guardan los títulos de la tabla que se quiere almacenar


tablehead = []


for header in soup.body.thead.tr.children:
    tablehead.append(header.text)


# En este apartado se limpian algunos datos erróneos como el dato nulo encontrado en 'Volume' del día 25/12/2019, que se cambia de '-' a 0.
# También se modifica el registro que contiene la fecha 17/12/2019 duplicada con un valor "Dividend".


if len(soup.body.tbody.find_all(text='-')) > 0:
    for dato in soup.body.tbody.find_all(text='-'):
        codigo = dato.find_parent('td')['data-reactid']
        soup.body.tbody.find('td', attrs={'data-reactid': codigo}).string = '0'


if len(soup.body.tbody.find_all(text='Dividend')) > 0:
    for registro in soup.body.tbody.find_all(text='Dividend'):
        registro.find_parent('tr').extract()


# Se crea una lista vacía y se vuelcan los datos extraídos en ella. Cada dato está unido a la etiqueta td (table data) y pueden contener tanto fechas como cotizaciones


datos = []
for dato in soup.body.tbody.find_all('td'):
    datos.append(dato.string)
```

Autores:

- Enrique Javier Andrés Orera
- Rubén Silva Marín

```
# Se cambia la forma de la lista para que coincida con las columnas volcadas anteriormente en 'tablehead'.

# Posteriormente se crea un pandas DataFrame con tablehead como columnas y se establece la columna Date como nombre de los índices.


datos = np.reshape(datos, (int(len(datos) / len(tablehead)), len(tablehead)))

datos = pd.DataFrame(datos, columns=tablehead)


# Se cambia formato de fechas ejemplo Apr 09, 2020 a 2020-4-09


for i in range(len(datos['Date'])):

    datos['Date'][i] = datetime.datetime.strptime(str(datos['Date'][i]), '%b %d, %Y').strftime('%Y-%m-%d')


datos = datos.set_index('Date', verify_integrity=True)


# En esta parte se deben cambiar las comas en la variable 'Volume', por puntos para poder transofrmar el formato numérico con el método pd.to_numeric de pandas.


datos = datos.replace(to_replace=r',', value='', regex=True)


datos = datos.apply(pd.to_numeric)


# Se exportan los datos en un archivo .CSV


datos.to_csv(ticker + '.csv')


# Se coloca la fecha en una columna del dataframe para poder realizar el gráfico


datos.reset_index(inplace=True, drop=False)


# Se ordena por fecha


datos = datos.sort_values('Date')


# Se crea un gráfico con los datos para mostrar como imagen descriptiva.


fig = go.Figure(data=go.Ohlc(x=datos['Date'],
                             open=datos['Open'],
                             high=datos['High'],
                             low=datos['Low'],
```

Autores:

- Enrique Javier Andrés Orera
- Rubén Silva Marín

```
                close=datos['Close*']))
fig.update(layout_xaxis_rangeslider_visible=False)

# Se añaden los títulos del gráfico

fig.update_layout(title=ticker, yaxis_title='Cotización')

# Se graba el gráfico html

fig.write_html(ticker + '.html')
```

10. Dataset. Publicación del dataset en formato CSV en Zenodo con una pequeña descripción.

Evolución de las cotizaciones de Telefónica durante los últimos cinco años.

Dataset con las cotizaciones de Telefónica (OHLC) de los últimos cinco años que tiene como objetivo principal servir de base para comprender cuál, en términos de cantidad y dinero, ha sido el impacto que ha tenido la pandemia Covid-19 en dichas cotizaciones.

Como quiera que Yahoo establece restricciones a la difusión de la información de la página de Yahoo Finance u otras entidades asociadas, el dataset se ha subido a zenodo con acceso restringido, pudiendo acceder el profesorado de la UOC.

El dataset se ha elaborado en el contexto del Máster Universitario en Ciencia de Datos de la Universitat Oberta de Catalunya, con objetivos educativos y no comerciales.

DOI 10.5281/zenodo.3751799

CONTRIBUCIONES	FIRMAS
Investigación previa	RSM, EJAO
Redacción de las respuestas	RSM, EJAO
Desarrollo código	RSM, EJAO

Autores:
- Enrique Javier Andrés Orera
- Rubén Silva Marín