Elaboración de webscraper

El primer paso es importar las librerías necesarias para la creación de este

```python
import pandas as pd
from bs4 import BeautifulSoup
from urllib.request import urlopen
import urllib.request
import requests
import time
from multiprocessing import Process, Queue, Pool
import threading
import sys
import numpy as np
import re
#from random_user_agent.user_agent import UserAgent
from random_user_agent.params import SoftwareName, OperatingSystem
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from fake_useragent import UserAgent
from selenium.webdriver.chrome.options import Options
import pandasql as ps
from IPython.display import display,HTML
from datetime import date
from datetime import datetime
import matplotlib
import matplotlib.pyplot as plt
```

El siguiente paso es crear las funciones que permitirán las búsquedas en los sitios web agregando las url, descargando un chromedriver y poniendo la dirección de donde se encuentra en el equipo

```python
def func_seminuevos(marca):
    path ="/Users/anethenlil/Downloads/chromedriver" # carga del web driver (asignar ruta donde se encuentra el driver)
    driver=webdriver.Chrome(path)

    time.sleep(5)
    url="https://www.cars.com/shopping/results/?dealer_id=&keyword="+marca
    driver.get(url) # instruccion de obtener url parametrizada
    time.sleep(10)
```

Lo siguiente es crear las tablas donde se guardará la información (nombre, precio, etc.) en la misma función con la clase que las contiene en el sitio web

```python
    productos=driver.find_elements_by_class_name("vehicle-card-main.js-gallery-click-card")

    # asignacion de nombres
    lista_nombres=[]
    for i in range(0,len(productos)):
        try:
            lista_nombres.append(productos[i].find_elements_by_class_name("vehicle-card-link.js-gallery-click-link")[0].text)
        except:
            lista_nombres.append(np.nan)
```

Al final de la función lo que se hace es guardar en columnas la información del paso anterior de esta forma y agregar la fecha que marcará el día en el que se hace la consulta

```
    today= date.today()

    df_seminuevos =pd.DataFrame(columns=["MODELO","PRECIO","MENSUALIDAD"])
    df_seminuevos["MODELO"] = lista_nombres
    df_seminuevos["PRECIO"] = lista_precios
    df_seminuevos["MENSUALIDAD"] = lista_mens
    df_seminuevos["SITIO"] = "cars.com"
    df_seminuevos["FECHA"] = str(today)

    driver.quit()

    return df_seminuevos
```

Sigue la parte de la creación de las tablas, en este caso mandamos a llamar a la función que creamos en un inicio y como parámetro escribimos la búsqueda (“honda”), luego el .insert que aparece es para crear una nueva columna (“MARCA”) con los valores de (“HONDA”), así con los 3 productos que queremos buscar

```
prod1 = func_seminuevos("honda")
time.sleep(10)
prod1.insert(1,"MARCA",'HONDA')
prod1
```

	MODELO	MARCA	PRECIO	MENSUALIDAD	SITIO	FECHA
0	2020 Honda HR-V Touring	HONDA	$29,997	$421 est./mo.*	cars.com	2022-12-15
1	2021 Honda Insight Touring	HONDA	$29,830	$418 est./mo.*	cars.com	2022-12-15
2	2019 Honda Pilot Touring 8-Passenger	HONDA	$31,044	$435 est./mo.*	cars.com	2022-12-15
3	2017 Honda Accord EX w/Honda Sensing	HONDA	$22,990	$322 est./mo.*	cars.com	2022-12-15
4	2013 Honda Pilot Touring	HONDA	$17,833	$250 est./mo.*	cars.com	2022-12-15

Después concatenamos las tablas que hayamos creado, las guardamos en un DataFrame y la parte de abajo elimina renglones donde se encuentren valores vacíos, en este caso en (“MENSUALIDAD”)

```
df_seminuevos_final = pd.concat([prod1,prod2,prod3])
df_seminuevos_final = df_seminuevos_final[df_seminuevos_final.MENSUALIDAD!=""]
```

El paso que sigue se hace para que las consultas en sql no marquen errores, lo que se hace es eliminar los puntos, comas, símbolos en general, convertir las cadenas que son precios (“PRECIO” y “MENSUALIDAD”) a valores flotantes y en este caso hacer la conversión a pesos ya que están en dólares, así con las otras páginas que usemos

```
df_seminuevos_final.PRECIO = df_seminuevos_final.PRECIO.str.replace(",","")
df_seminuevos_final.PRECIO = df_seminuevos_final.PRECIO.str.replace("$","")
df_seminuevos_final.PRECIO = df_seminuevos_final.PRECIO.str.replace(".","")
```

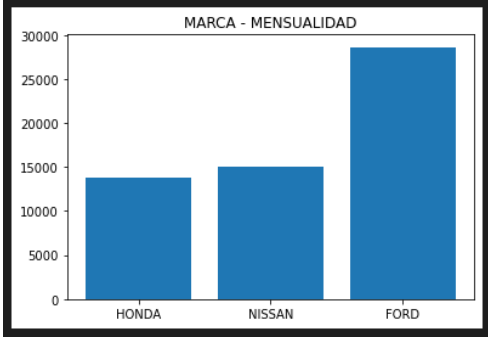
```
df_seminuevos_final.PRECIO = df_seminuevos_final.PRECIO.astype(float) # cast de datos
df_seminuevos_final.PRECIO = df_seminuevos_final.PRECIO*(19.53) #convertir de usd a mxn
```

Al final de todo este proceso concatenamos las últimas tablas que ya tienen todo el contenido

```
df_tabla_final = pd.concat([df_seminuevos_final,df_usados_final,df_olx_final])
df_tabla_final
```

Luego para crear las gráficas es necesario hacer uso de la librería matplotlib, aquí se comparan las medias de las mensualidades de las marcas en la tabla final

```
plt.bar(df_tabla_final['MARCA'],df_tabla_final['MENSUALIDAD']) #gráfica que muestra la tendencia de las mensualidades
plt.title('MARCA - MENSUALIDAD')
plt.show()
```



Siguiendo ya solo se hacen consultas de sql para conocer valores específicos en la tabla como lo es así

```
1  ps.sqldf("SELECT * FROM df_olx_final where PRECIO>500000") #2
```

	MODELO	MARCA	PRECIO	MENSUALIDAD	SITIO	FECHA
0	2022, Honda Insight	HONDA	592000.0	11517.0	olxautos.com	2022-12-15
1	2018, Honda Pilot	HONDA	573000.0	11147.0	olxautos.com	2022-12-15
2	2018, Honda Odyssey	HONDA	708000.0	13774.0	olxautos.com	2022-12-15
3	2019, Honda Insight	HONDA	534000.0	10389.0	olxautos.com	2022-12-15
4	2019, Honda CR-V	HONDA	518000.0	10077.0	olxautos.com	2022-12-15
5	2019, Honda CR-V	HONDA	518000.0	10077.0	olxautos.com	2022-12-15
6	2019, Honda CR-V	HONDA	533000.0	10369.0	olxautos.com	2022-12-15
7	2019, Nissan X-Trail	NISSAN	513530.0	9990.0	olxautos.com	2022-12-15
8	2019, Ford Edge	FORD	573000.0	11147.0	olxautos.com	2022-12-15
9	2019, Ford Edge	FORD	573000.0	11147.0	olxautos.com	2022-12-15
10	2019, Ford Fusion	FORD	533000.0	10369.0	olxautos.com	2022-12-15
11	2021, Ford Escape	FORD	665000.0	12937.0	olxautos.com	2022-12-15
12	2021, Ford ford-bronco	FORD	740000.0	14396.0	olxautos.com	2022-12-15

Al final o en el momento que ya se crea la última tabla se puede guardar como Excel o csv

```
1  df_tabla_final.to_excel("df_tabla_final.xlsx",index=False)
```