Automation Collection and Analysis Commodities Quotations

About:

Development of a script (algorithm) aimed at automating the process of obtaining commodities information.

Proposal:

Develop a script that automates the collection of commodities quotations, analyze if the value is below the ideal price defined in the dataset for purchase, and, if necessary, update it with buying recommendations. The final result will be exported in a new .xlsx file.

The website used for collecting quotations was "**Melhor Câmbio**".

Repository Structure:

- **data:** You will find the **.xlsx** file of the base dataset for research and the dataset generated at the end of the script.
- **img:** This is where the screenshots obtained during the analysis can be found.
- **script:** This directory contains the developed script.
- **readme_translated:** This repository contains the **PDF** with the report translated into English.

Language Used:

Libraries Used:

Methodology:

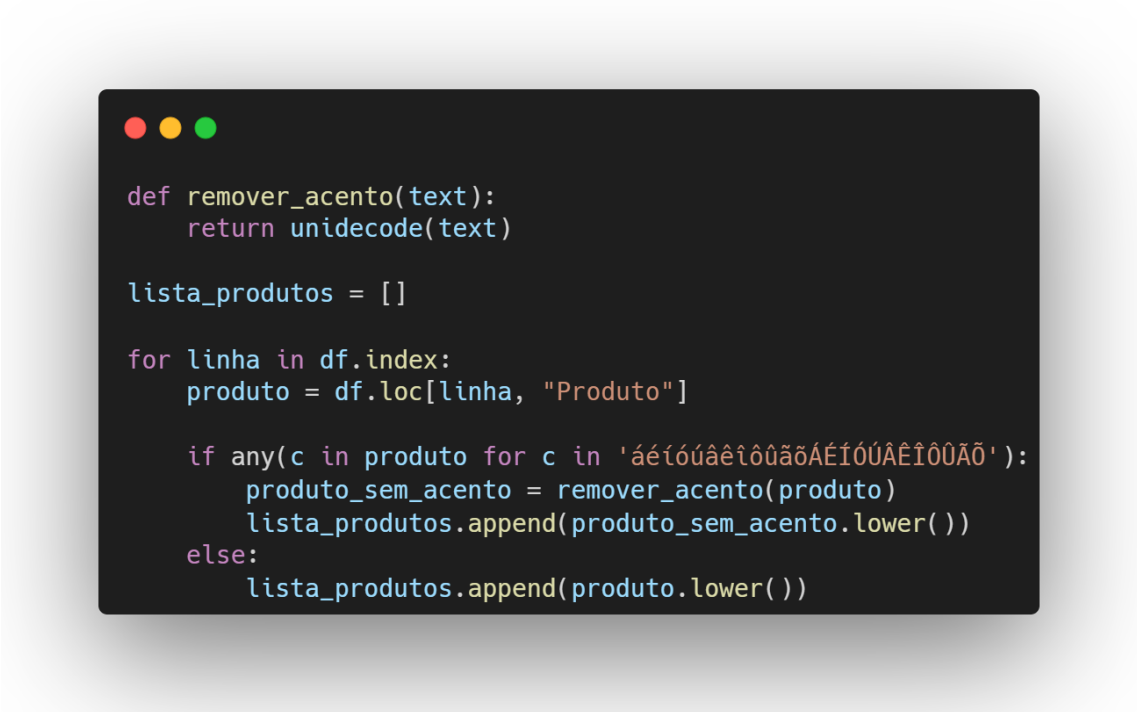
Initially, the **Pandas** library was used to import and read the dataset.

	Produto	Preço Ideal	Preço Atual	Comprar
0	Milho	85.32	NaN	NaN
1	Soja	163.59	NaN	NaN
2	Boi	282.20	NaN	NaN
3	Petróleo	424.37	NaN	NaN
4	Algodão	497.76	NaN	NaN
5	Açúcar	136.23	NaN	NaN
6	Café	1092.87	NaN	NaN
7	Ouro	321.77	NaN	NaN
8	Trigo	1549.11	NaN	NaN
9	Tilápia	9.05	NaN	NaN

It was possible to identify the commodities for which current values would be collected and the ideal prices.

Next, the treatment of product names was done, where some had the first letter in uppercase or contained accents. To perform this task, the **Unidecode** library was used.

First, a function was created to remove characters from the text. A **for** loop was used to iterate through the rows, requiring a conditional structure.



```
def remover_acento(text):
    return unidecode(text)

lista_produtos = []

for linha in df.index:
    produto = df.loc[linha, "Produto"]

    if any(c in produto for c in 'áéíóúâêîôûãõÁÉÍÓÚÂÊÎÔÛÃÕ'):
        produto_sem_acento = remover_acento(produto)
        lista_produtos.append(produto_sem_acento.lower())
    else:
        lista_produtos.append(produto.lower())
```

The code snippet **if any(c in product for c in 'áéíóúâêîôûãõÁÉÍÓÚÂÊÎÔÛÃÕ')** checks if any of the accented characters from the list are present in the variable "**product**". If at least one of these characters is present, the **any(...)** expression will evaluate as **true**; otherwise, it will evaluate as **false**.

Finally, the **.append** function was used to create a new list containing the necessary modifications.

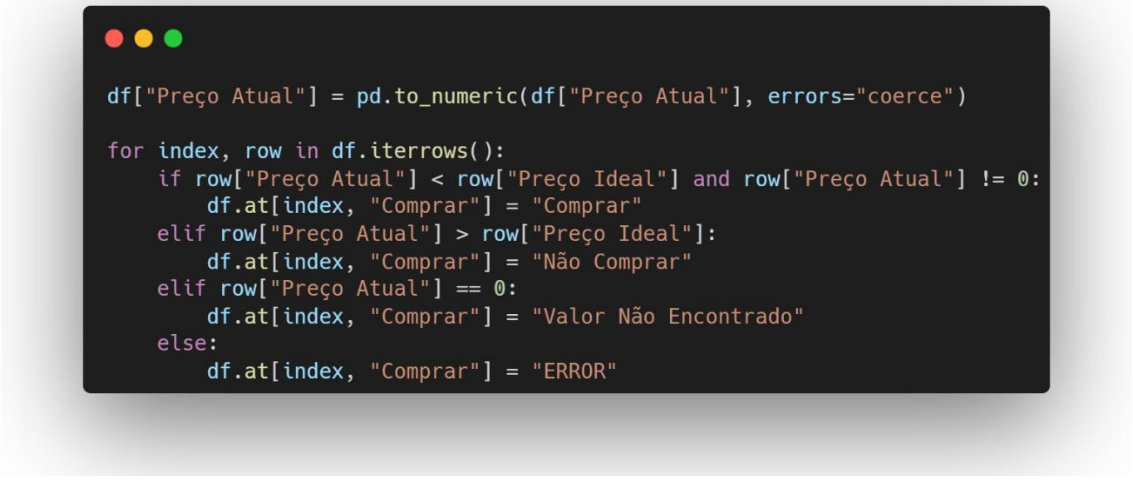
The automated collection was implemented using the **Selenium** framework's **Webdriver**.

```
navegador = webdriver.Chrome()

for linha, produto in enumerate(lista_produtos):

    link = f"https://www.melhorcambio.com/{produto}-hoje"

    navegador.get(link)
    preco_produto = navegador.find_element("xpath", '//*[@id="comercial"]').get_attribute("value")
    preco_produto = preco_produto.replace(".", "").replace(",", ".")

    df.loc[linha, "Preço Atual"] = preco_produto

navegador.quit()
```

Xpath was used as a reference point to extract the "**value**" attribute, and it was necessary to replace commas with periods to adhere to the **Python** standard.

The use of the **enumerate** function allowed obtaining the index and value of the product, thus enabling the DataFrame to be updated during the loop iteration.

Processing of the "**Current Price**" column and the subsequent filling of the "**Buy**" column with essential information. This final step involved applying the necessary operations to ensure the accuracy of the appropriate insertions in the **"Buy"** column.



```
df["Preço Atual"] = pd.to_numeric(df["Preço Atual"], errors="coerce")

for index, row in df.iterrows():
    if row["Preço Atual"] < row["Preço Ideal"] and row["Preço Atual"] != 0:
        df.at[index, "Comprar"] = "Comprar"
    elif row["Preço Atual"] > row["Preço Ideal"]:
        df.at[index, "Comprar"] = "Não Comprar"
    elif row["Preço Atual"] == 0:
        df.at[index, "Comprar"] = "Valor Não Encontrado"
    else:
        df.at[index, "Comprar"] = "ERROR"
```

The final result was exported in a new **.xlsx** file.

	Produto	Preço Ideal	Preço Atual	Comprar
0	Milho	85.32	52.97	Comprar
1	Soja	163.59	142.70	Comprar
2	Boi	282.20	237.35	Comprar
3	Petróleo	424.37	421.57	Comprar
4	Algodão	497.76	404.06	Comprar
5	Açúcar	136.23	135.68	Comprar
6	Café	1092.87	843.74	Comprar
7	Ouro	321.77	305.52	Comprar
8	Trigo	1549.11	0.00	Valor Não Encontrado
9	Tilápia	9.05	9.41	Não Comprar

Note: The values obtained are related to the day of the publication of this repository.

Conclusion:

Throughout this process of automating the collection and analysis of commodities quotations, several steps were taken to optimize and expedite data acquisition.

The final step, which included processing the "**Current Price**" column and filling the "**Buy**" column with the necessary information, added a strategic element to the analysis.

The result of this automation and analysis process was a new **.xlsx** file containing relevant information about commodities quotations and purchase markings. This approach not only saved time and effort but also

contributed to more effective decision-making. Automation proved to be a valuable tool in simplifying and expediting processes, boosting efficiency and accuracy in management and analysis.