

Programa

```
# -*- coding: utf-8 -*-
```

```
"""calculateSales.py
```

Automatically generated by Colab.

Original file is located at

https://colab.research.google.com/drive/1okpv7m4Y_MEyEsfSoB5GnSkEcN7SYIeE

Calcula el costo total de ventas tomando como base el catálogo de precios.

Este programa toma dos archivos JSON como entrada:

1. priceCatalogue.json - Contiene información de los precios de productos.
2. salesRecord.json - Contiene un registro de las ventas realizadas.

El resultado se muestra en pantalla y se guarda en SalesResults.txt.

```
"""
```

```
import json
```

```
import sys
```

```
import time
```

```
import os
```

```
import glob
```

```
from google.colab import drive
```

```

drive.mount('/content/drive', force_remount=True)

# Definir constantes para las rutas de archivos

PRODUCTS_FILE = (

    "/content/drive/MyDrive/Calidad_software/Semana5/TC1.ProductList.json"

)

SALES_FILES_PATTERN = (

    "/content/drive/MyDrive/Calidad_software/Semana5/TC*.Sales.json"

)

RESULT_FILE = (

    "/content/drive/MyDrive/Calidad_software/Semana5/SalesResults.txt"

)


def load_json(file_path):

    """

    Carga un archivo JSON y maneja errores.

    """

    try:

        with open(file_path, "r", encoding="utf-8") as file:

            return json.load(file)

    except FileNotFoundError:

        print(f"Error: No se encontró el archivo {file_path}.")

    except json.JSONDecodeError:

        print(f"Error: {file_path} no tiene un formato JSON válido.")

    return None

```

```

def build_price_catalogue(product_list):

    """

    Crea un diccionario de precios basado en el título

    del producto.

    """

    return {product["title"]: product["price"] for

            product in product_list}


def calculate_total_sales(price_catalogue, sales_records):

    """

    Calcula el costo total de ventas basado en el catálogo de precios.

    """

    total_sales = 0.0

    detailed_sales = []

    errors = []

    for sale in sales_records:

        product_name = sale.get("Product")

        quantity = sale.get("Quantity")

        if product_name not in price_catalogue:

            errors.append(

                f"Producto no encontrado en catálogo: {product_name}"

            )

```

```
continue
```

```
if not isinstance(quantity, (int, float)) or quantity <= 0:
```

```
    errors.append(
```

```
        f"Cantidad inválida para producto {product_name}: {quantity}"
```

```
    )
```

```
    continue
```

```
price = price_catalogue[product_name]
```

```
total_cost = price * quantity
```

```
total_sales += total_cost
```

```
detailed_sales.append(
```

```
    f"Producto: {product_name}, Cantidad: {quantity}, "
```

```
    f"Precio unitario: ${price:.2f}, Costo total: ${total_cost:.2f}"
```

```
    )
```

```
return total_sales, detailed_sales, errors
```

```
def main():
```

```
    """Función principal del programa."""
```

```
    start_time = time.time()
```

```
    # Cargar lista de productos y construir catálogo de precios
```

```
    product_data = load_json(PRODUCTS_FILE)
```

```
    if product_data is None:
```

```

    sys.exit(1)

price_catalogue = build_price_catalogue(product_data)

# Cargar todas las ventas de los archivos TCx.Sales.json

sales_data = []

for sales_file in glob.glob(SALES_FILES_PATTERN):

    sales = load_json(sales_file)

    if sales:

        sales_data.extend(sales)

if not sales_data:

    print("Error: No se encontraron datos de ventas válidos.")

    sys.exit(1)

# Calcular ventas totales

total_sales, detailed_sales, errors = calculate_total_sales(

    price_catalogue, sales_data

)

elapsed_time = time.time() - start_time

# Generar informe

result_text = (

    f"Costo total de ventas: ${total_sales:.2f}\n\n"

    "Detalles de ventas:\n" + "\n".join(detailed_sales) +

    f"\n\nTiempo de ejecución: {elapsed_time:.4f} segundos\n"

```

```
)
```

if errors:

```
result_text += "\nErrores encontrados:\n" + "\n".join(errors) + "\n"
```

```
print(result_text)
```

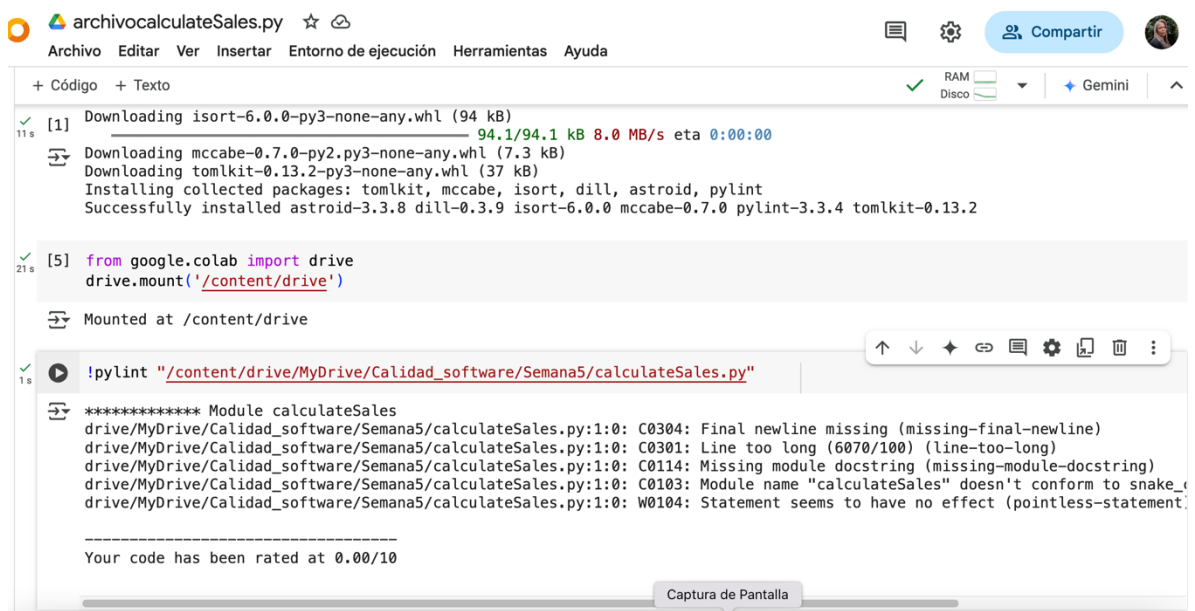
with open(RESULT_FILE, "w", encoding="utf-8") as result_file:

```
result_file.write(result_text)
```

```
if __name__ == "__main__":
```

```
main()
```

Al Ejecutar el comando Pylint para las validaciones dinámicas se encontró:



The screenshot shows a Google Colab notebook interface. At the top, the file name is 'archivocalculateSales.py'. The menu bar includes 'Archivo', 'Editar', 'Ver', 'Insertar', 'Entorno de ejecución', 'Herramientas', and 'Ayuda'. On the right, there are icons for chat, settings, a 'Compartir' button, and a user profile. The notebook has two tabs: 'Código' and 'Texto'. The first cell, labeled '[1]', shows the output of a pip install command, listing various packages being downloaded and installed. The second cell, labeled '[5]', shows the output of a 'drive.mount' command, indicating that the drive is mounted at '/content/drive'. The third cell, labeled '[!]', shows the output of a Pylint command executed on the file '/content/drive/MyDrive/Calidad_software/Semana5/calculateSales.py'. The Pylint output lists several errors: 'C0304: Final newline missing (missing-final-newline)', 'C0301: Line too long (6070/100) (line-too-long)', 'C0114: Missing module docstring (missing-module-docstring)', 'C0103: Module name "calculateSales" doesn't conform to snake_case', and 'W0104: Statement seems to have no effect (pointless-statement)'. At the bottom, it says 'Your code has been rated at 0.00/10'. A 'Captura de Pantalla' button is visible at the very bottom.

```
archivocalculateSales.py ☆
Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

+ Código + Texto

[1] Downloading isort-6.0.0-py3-none-any.whl (94 kB)
    94.1/94.1 kB 8.0 MB/s eta 0:00:00
    Downloading mccabe-0.7.0-py2.py3-none-any.whl (7.3 kB)
    Downloading tomlkit-0.13.2-py3-none-any.whl (37 kB)
    Installing collected packages: tomlkit, mccabe, isort, dill, astroid, pylint
    Successfully installed astroid-3.3.8 dill-0.3.9 isort-6.0.0 mccabe-0.7.0 pylint-3.3.4 tomlkit-0.13.2

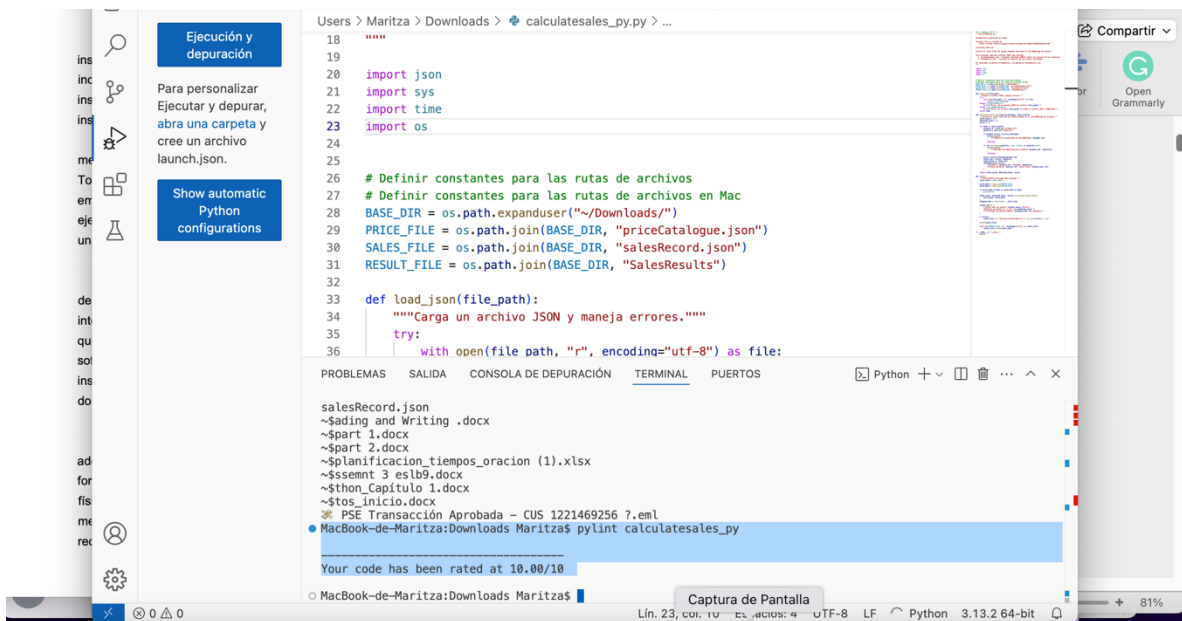
[5] from google.colab import drive
    drive.mount('/content/drive')
Mounted at /content/drive

!pylint "/content/drive/MyDrive/Calidad_software/Semana5/calculateSales.py"
***** Module calculateSales
drive/MyDrive/Calidad_software/Semana5/calculateSales.py:1:0: C0304: Final newline missing (missing-final-newline)
drive/MyDrive/Calidad_software/Semana5/calculateSales.py:1:0: C0301: Line too long (6070/100) (line-too-long)
drive/MyDrive/Calidad_software/Semana5/calculateSales.py:1:0: C0114: Missing module docstring (missing-module-docstring)
drive/MyDrive/Calidad_software/Semana5/calculateSales.py:1:0: C0103: Module name "calculateSales" doesn't conform to snake_case
drive/MyDrive/Calidad_software/Semana5/calculateSales.py:1:0: W0104: Statement seems to have no effect (pointless-statement)

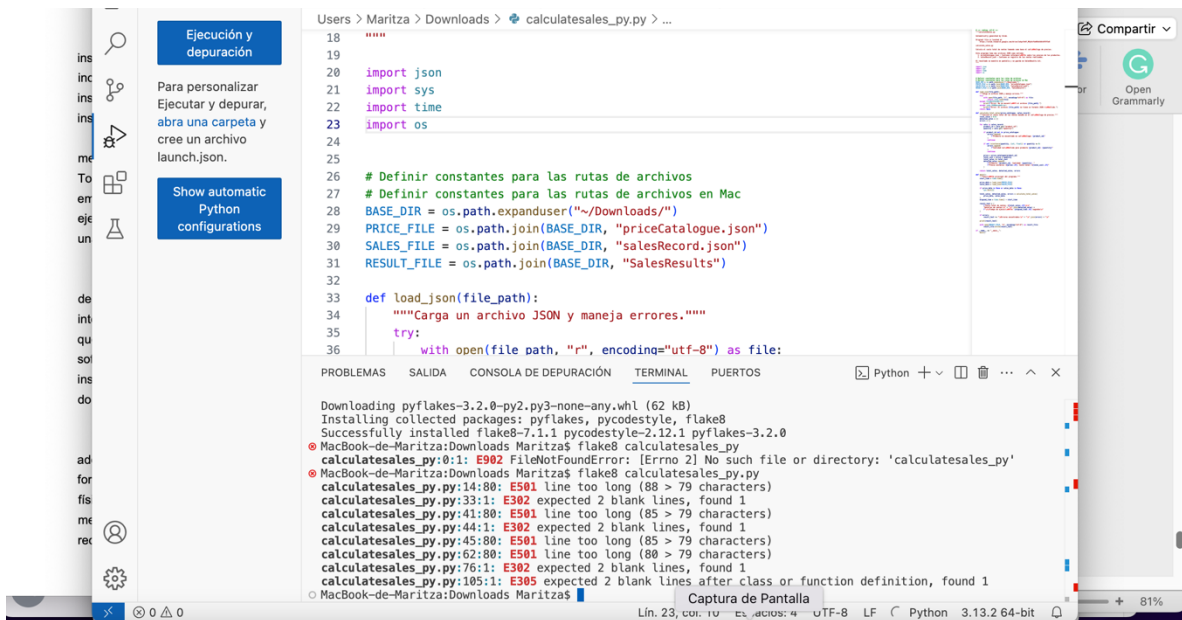
Your code has been rated at 0.00/10

Captura de Pantalla
```

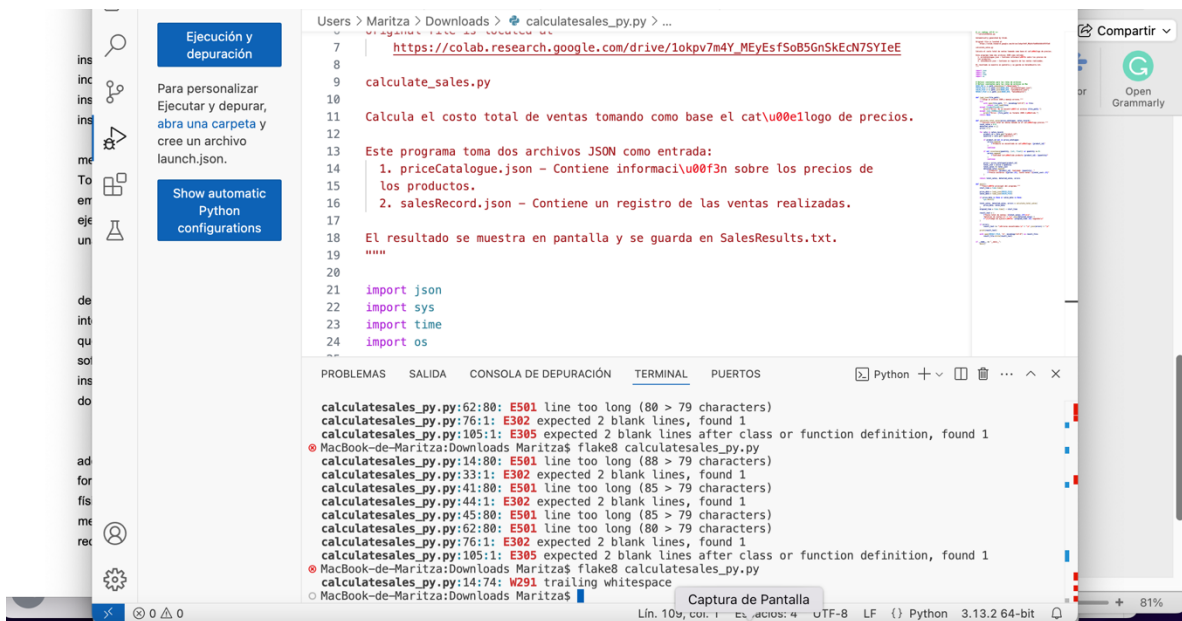
En colab no me ejecuta correctamente, lo realice en Visual y me salió todo ok



Al ejecutar Flake8 salio por visual:



Se ajustaron cosas se subsanaron, pero sale otro error



Ultima ejecución eliminando espacios y reduciendo el texto

