

México, 11 de febrero de 2024

Actividad 5.2 | Ejercicio de programación 2

Enlace al repositorio de GitHub: https://github.com/ErickHCerecedo/A01066428_A5.2

El siguiente archivo contiene la solución implementada en Python al ejercicio de programación 2, en el que se reporta el proceso de creación del ejercicio desde la ejecución de revisión de gramática y estilo usando PyLint, análisis de errores empleando Flake y por último la ejecución de pruebas unitarias.

Problem 1: Compute Sales	
Description	<p>Requirement 1. The program shall be invoked from a command line. The program shall receive two files as parameters. The first file will contain information in a JSON format about a catalogue of prices of products. The second file will contain a record for all sales in a company.</p> <p>Requirement 2. The program shall compute the total cost for all sales included in the second JSON archive. The results shall be print on a screen and on a file named SalesResults.txt. The total cost should include all items in the sale considering the cost for every item in the first file.</p> <p>The output must be human readable, so make it easy to read for the user.</p> <p>Requirement 3. The program shall include the mechanism to handle invalid data in the file. Errors should be displayed in the console and the execution must continue.</p> <p>Requirement 4. The name of the program shall be computeSales.py</p> <p>Requirement 5. The minimum format to invoke the program shall be as follows: python computeSales.py priceCatalogue.json salesRecord.json</p> <p>Requirement 6. The program shall manage files having from hundreds of items to thousands of items.</p> <p>Requirement 7. The program should include at the end of the execution the time elapsed for the execution and calculus of the data. This number shall be included in the results file and on the screen.</p> <p>Requirement 8. Be compliant with PEP8.</p>

1. Implementación inicial del código:

The screenshot displays a Windows desktop environment. The primary focus is a code editor window titled 'computesales.py' containing Python code for a sales management application. The code includes a `load_json` function to load data from JSON files, a `compute_total_cost` function to calculate the total cost of sales, and a `process_sales` function to process sales records. The code uses `try-except` blocks to handle file not found and JSON decoding errors.

In the background, a terminal window is open, showing a series of Git commands and their outputs. The commands include `git init`, `git add`, `git commit`, `git remote add`, `git push`, `git branch`, `git remote add origin`, and `git push`. The output shows the repository being initialized, files being added and committed, and the repository being pushed to a remote location.

The taskbar at the bottom of the screen shows the Start button, a search bar, and several open applications including the code editor, terminal, and file explorer. The system tray in the bottom right corner shows the date and time as 05:11 on 11/2/2020.

2. Análisis de errores de PyLint (PEP8):

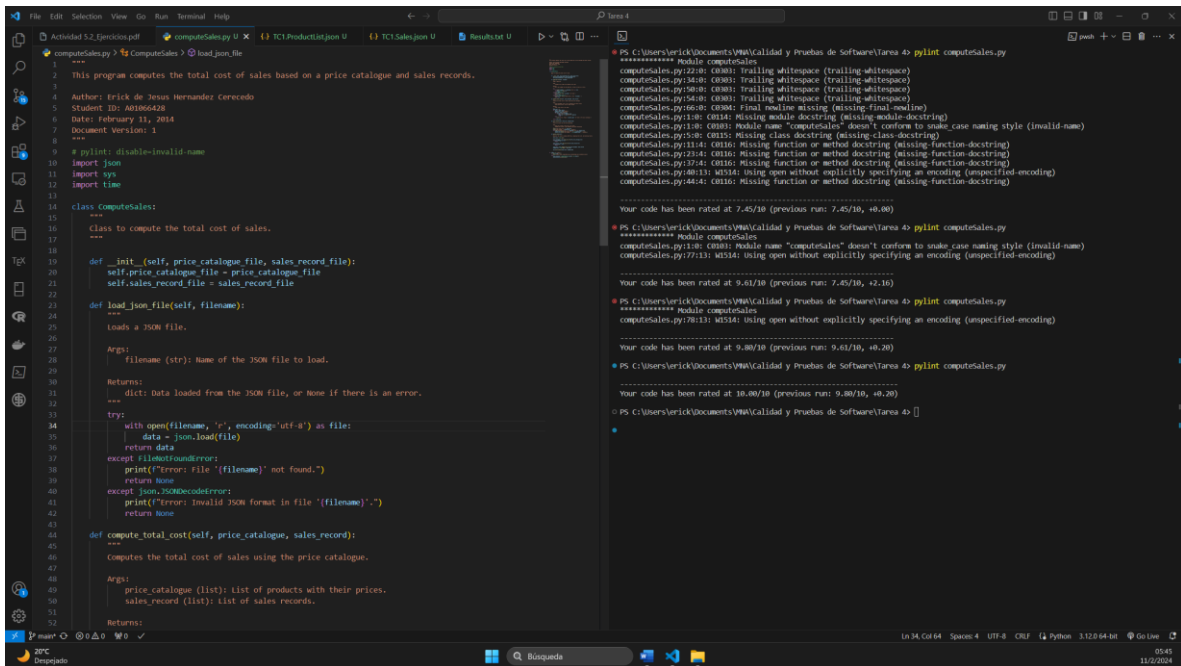
The screenshot displays a Windows desktop environment. The primary focus is a code editor window titled 'computesales.py', which contains the following Python code:

```
1 import json
2 import sys
3 import time
4
5 class Computesales:
6
7     def __init__(self, price_catalogue_file, sales_record_file):
8         self.price_catalogue_file = price_catalogue_file
9         self.sales_record_file = sales_record_file
10
11     def load_json_file(self, filename):
12         try:
13             with open(filename, 'r', encoding='utf-8') as file:
14                 data = json.load(file)
15                 return data
16         except FileNotFoundError:
17             print("Error: File '{filename}' not found.")
18             return None
19         except json.JSONDecodeError:
20             print("Error: Invalid JSON format in file '{filename}'.")
21             return None
22
23     def compute_total_cost(self, price_catalogue, sales_record):
24         total_cost = 0
25         for sale in sales_record:
26             product_name = sale['product']
27             quantity = sale['quantity']
28             for item in price_catalogue:
29                 if item['title'] == product_name:
30                     total_cost += item['price'] * quantity
31                     break
32             else:
33                 print(f"Error: Product '{product_name}' not found in the price catalogue.")
34
35         return total_cost
36
37     def print_results(self, total_cost, elapsed_time):
38         print(f"Total sales cost: $", total_cost)
39         print(f"Time elapsed: ", elapsed_time, " seconds")
40         with open('salesresults.txt', 'w') as results_file:
41             results_file.write(f"Total sales cost: $ + str(total_cost) + '\n')
42             results_file.write(f"Time elapsed: " + str(elapsed_time) + " seconds\n")
43
44     def process_sales(self):
45         start_time = time.time()
46
47         price_catalogue = self.load_json_file(self.price_catalogue_file)
48         if price_catalogue is None:
49             return
50
51         sales_record = self.load_json_file(self.sales_record_file)
52         if sales_record is None:
```

To the right of the code editor, a terminal window is open, displaying a list of linting errors from 'pylint computesales.py'. The errors include issues such as 'missing module docstring', 'missing class docstring', 'missing function docstring', and 'no explicit encoding specified'. The terminal also shows a score of 7.45/10 and a message indicating the code has been rated.

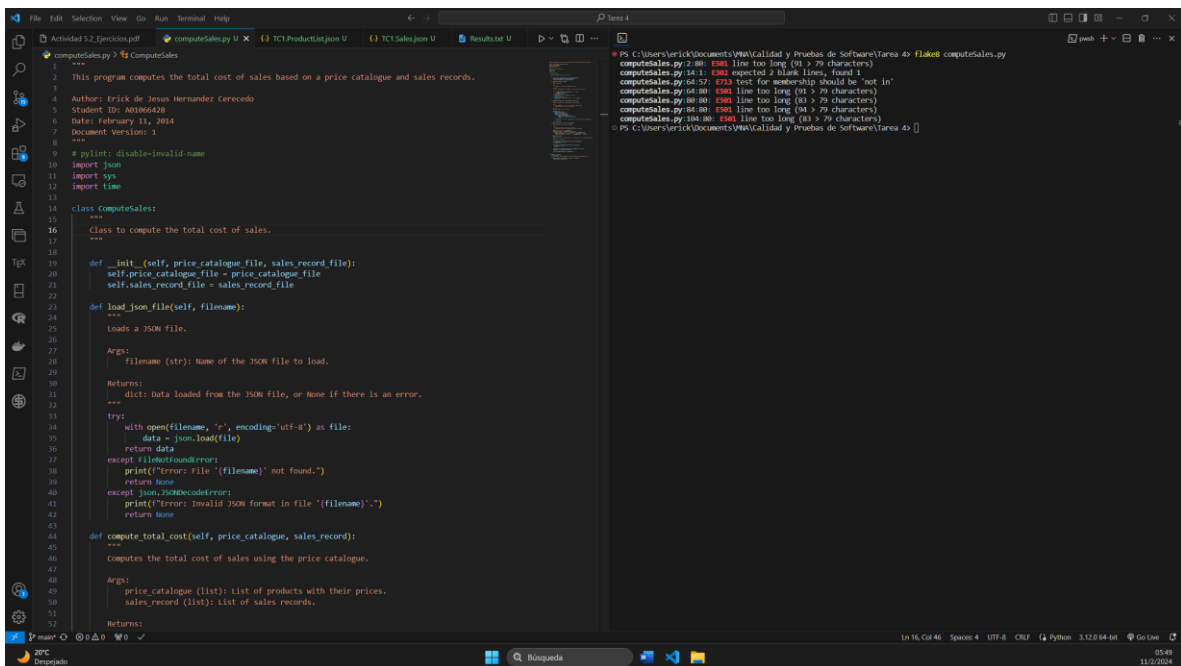
At the bottom of the screen, a taskbar is visible with the Windows Start button, a search bar labeled 'Busqueda', and several open application icons. The system tray on the right shows the date and time as '11/2/2021' and '11:20 AM'.

Corrección de errores:



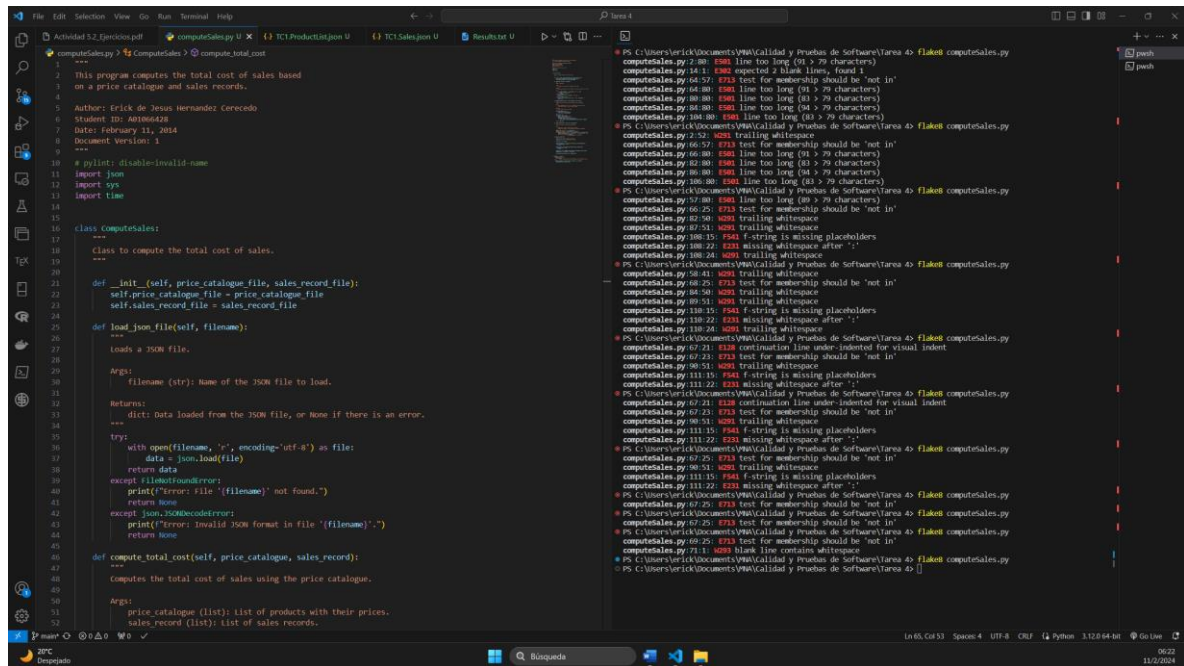
```
File Edit Selection View Go Run Terminal Help
computeSales.py x TC1.ProductoListJson U TC1.SalesJson U Resultados U
computeSales.py 3 ComputeSales > load_json_file
1 """
2 This program computes the total cost of sales based on a price catalogue and sales records.
3
4 Author: Erick de Jesus Hernandez Cerecedo
5 Student ID: A01806420
6 Date: February 11, 2014
7 Document Version: 1
8 """
9 # pylint: disable=invalid-name
10 import json
11 import sys
12 import time
13
14 class ComputeSales:
15     """
16     Class to compute the total cost of sales.
17     """
18
19     def __init__(self, price_catalogue_file, sales_record_file):
20         self.price_catalogue_file = price_catalogue_file
21         self.sales_record_file = sales_record_file
22
23     def load_json_file(self, filename):
24         """
25         Loads a JSON file.
26
27         Args:
28             filename (str): Name of the JSON file to load.
29
30         Returns:
31             dict: Data loaded from the JSON file, or None if there is an error.
32         """
33         try:
34             with open(filename, 'r', encoding='utf-8') as file:
35                 data = json.load(file)
36             return data
37         except FileNotFoundError:
38             print("Error: File '{filename}' not found.")
39             return None
40         except json.JSONDecodeError:
41             print("Error: Invalid JSON format in file '{filename}'.")
42             return None
43
44     def compute_total_cost(self, price_catalogue, sales_record):
45         """
46         Computes the total cost of sales using the price catalogue.
47
48         Args:
49             price_catalogue (list): List of products with their prices.
50             sales_record (list): List of sales records.
51
52         Returns:
53             float: Total cost of sales.
54         """
55
56 PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> pylint computeSales.py
57 =====
58 Module: computeSales
59 computeSales.py:22:0: CR001: Trailing whitespace (trailing-whitespace)
60 computeSales.py:24:0: CR001: Trailing whitespace (trailing-whitespace)
61 computeSales.py:26:0: CR001: Trailing whitespace (trailing-whitespace)
62 computeSales.py:28:0: CR001: Trailing whitespace (trailing-whitespace)
63 computeSales.py:30:0: CR001: Trailing whitespace (trailing-whitespace)
64 computeSales.py:32:0: CR001: Final newline missing (missing-final-newline)
65 computeSales.py:138: CR014: Missing module docstring (missing-module-docstring)
66 computeSales.py:138: CR001: Module name "computeSales" doesn't conform to snake case naming style (invalid-name)
67 computeSales.py:5:0: CR015: Missing class docstring (missing-class-docstring)
68 computeSales.py:14:0: CR016: Missing function or method docstring (missing-function-docstring)
69 computeSales.py:19:0: CR016: Missing function or method docstring (missing-function-docstring)
70 computeSales.py:23:0: CR016: Missing function or method docstring (missing-function-docstring)
71 computeSales.py:44:0: CR016: Missing function or method docstring (missing-function-docstring)
72 computeSales.py:77:13: W0344: Using open without explicitly specifying an encoding (unspecified-encoding)
73
74 Your code has been rated at 7.45/10 (previous run: 7.45/10, +0.00)
75
76 PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> pylint computeSales.py
77 =====
78 Module: computeSales
79 computeSales.py:138: CR001: Module name "computeSales" doesn't conform to snake case naming style (invalid-name)
80 computeSales.py:77:13: W0344: Using open without explicitly specifying an encoding (unspecified-encoding)
81
82 Your code has been rated at 9.61/10 (previous run: 7.45/10, +2.16)
83
84 PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> pylint computeSales.py
85 =====
86 Module: computeSales
87 computeSales.py:77:13: W0344: Using open without explicitly specifying an encoding (unspecified-encoding)
88
89 Your code has been rated at 9.80/10 (previous run: 9.61/10, +0.20)
90
91 PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> pylint computeSales.py
92 =====
93
94 Your code has been rated at 10.00/10 (previous run: 9.80/10, +0.20)
95
96 PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4>
97
98 Lin 34, Col 64 - Spaces: 4 - UTF-8 - CR/LF - Python - 3.12.0 64-bit - Go Live - 09:49 11/02/2024
```

3. Análisis de Errores Flake:



```
File Edit Selection View Go Run Terminal Help
computeSales.py x TC1.ProductoListJson U TC1.SalesJson U Resultados U
computeSales.py 3 ComputeSales > load_json_file
1 """
2 This program computes the total cost of sales based on a price catalogue and sales records.
3
4 Author: Erick de Jesus Hernandez Cerecedo
5 Student ID: A01806420
6 Date: February 11, 2014
7 Document Version: 1
8 """
9 # pylint: disable=invalid-name
10 import json
11 import sys
12 import time
13
14 class ComputeSales:
15     """
16     Class to compute the total cost of sales.
17     """
18
19     def __init__(self, price_catalogue_file, sales_record_file):
20         self.price_catalogue_file = price_catalogue_file
21         self.sales_record_file = sales_record_file
22
23     def load_json_file(self, filename):
24         """
25         Loads a JSON file.
26
27         Args:
28             filename (str): Name of the JSON file to load.
29
30         Returns:
31             dict: Data loaded from the JSON file, or None if there is an error.
32         """
33         try:
34             with open(filename, 'r', encoding='utf-8') as file:
35                 data = json.load(file)
36             return data
37         except FileNotFoundError:
38             print("Error: File '{filename}' not found.")
39             return None
40         except json.JSONDecodeError:
41             print("Error: Invalid JSON format in file '{filename}'.")
42             return None
43
44     def compute_total_cost(self, price_catalogue, sales_record):
45         """
46         Computes the total cost of sales using the price catalogue.
47
48         Args:
49             price_catalogue (list): List of products with their prices.
50             sales_record (list): List of sales records.
51
52         Returns:
53             float: Total cost of sales.
54         """
55
60 PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> flake8 computeSales.py
61 =====
62 computeSales.py:1:0: E302: line too long (91 > 79 characters)
63 computeSales.py:14:1: E300: expected 2 blank lines, found 1
64 computeSales.py:64:57: E713: Test for membership should be "not in"
65 computeSales.py:64:80: E301: line too long (93 > 79 characters)
66 computeSales.py:80:80: E301: line too long (80 > 79 characters)
67 computeSales.py:84:80: E301: line too long (84 > 79 characters)
68 computeSales.py:104:80: E301: line too long (83 > 79 characters)
69
70 PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4>
71
72 Lin 34, Col 64 - Spaces: 4 - UTF-8 - CR/LF - Python - 3.12.0 64-bit - Go Live - 09:49 11/02/2024
```

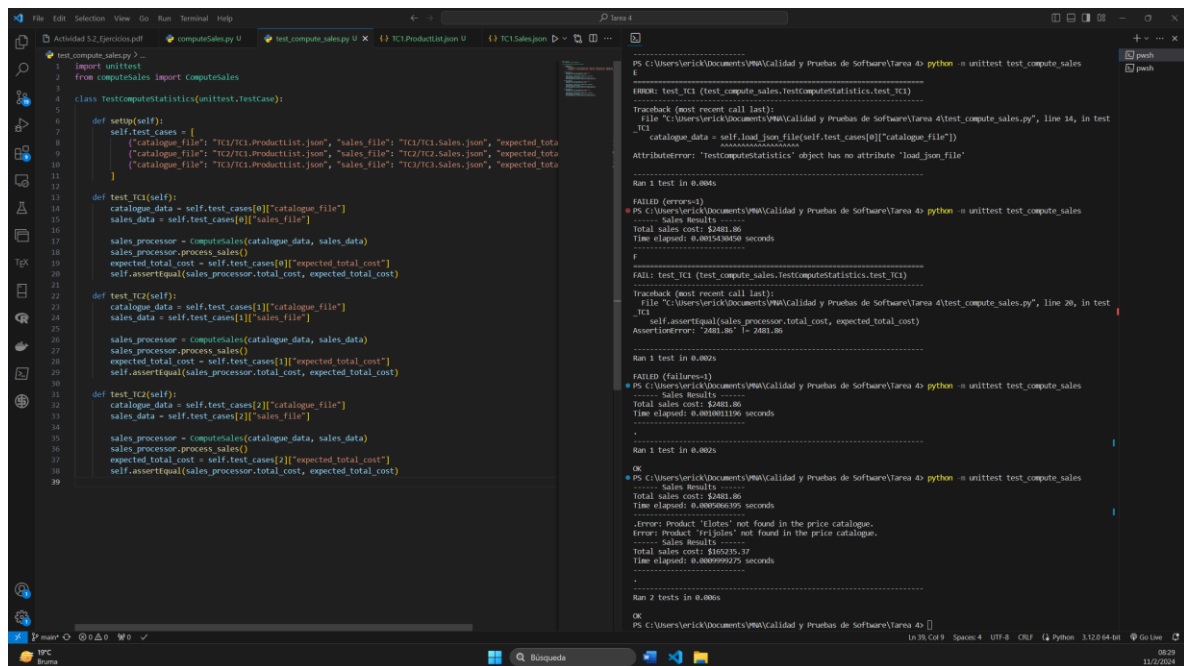
Corrección de errores con Flake8



```
1 # This program computes the total cost of sales based
2 # on a price catalogue and sales records.
3
4 Author: Erick de Jesus Hernandez Cerecedo
5 Student ID: A01066248
6 Date: February 11, 2024
7 Document version: 1
8
9 # pylint: disable=invalid-name
10
11 import json
12 import sys
13 import time
14
15
16 class Computales:
17     """
18     class to compute the total cost of sales.
19     """
20
21     def __init__(self, price_catalogue_file, sales_record_file):
22         self.price_catalogue_file = price_catalogue_file
23         self.sales_record_file = sales_record_file
24
25     def load_json_file(self, filename):
26         """
27         loads a JSON file.
28
29         Args:
30             filename (str): Name of the JSON file to load.
31
32         Returns:
33             dict: Data loaded from the JSON file, or None if there is an error.
34         """
35         try:
36             with open(filename, 'r', encoding='utf-8') as file:
37                 data = json.load(file)
38             return data
39         except FileNotFoundError:
40             print(f"Error: file '{filename}' not found.")
41             return None
42         except json.JSONDecodeError:
43             print(f"Error: invalid JSON format in file '{filename}'.")
44             return None
45
46     def compute_total_cost(self, price_catalogue, sales_record):
47         """
48         Computes the total cost of sales using the price catalogue.
49
50         Args:
51             price_catalogue (list): List of products with their prices.
52             sales_record (list): List of sales records.
53         """
```

```
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> flake8 computales.py
computales.py:1:1: E302 line too long (91 > 79 characters)
computales.py:14:1: E360 expected 2 blank lines, found 1
computales.py:14:25: E733 test for membership should be 'not in'
computales.py:14:40: E360 line too long (91 > 79 characters)
computales.py:16:40: E360 line too long (91 > 79 characters)
computales.py:16:80: E360 line too long (81 > 79 characters)
computales.py:16:80: E360 line too long (91 > 79 characters)
computales.py:106:80: E360 line too long (83 > 79 characters)
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> flake8 computales.py
computales.py:2:52: W291 trailing whitespace
computales.py:6:52: E733 test for membership should be 'not in'
computales.py:16:40: E360 line too long (91 > 79 characters)
computales.py:16:80: E360 line too long (81 > 79 characters)
computales.py:16:80: E360 line too long (91 > 79 characters)
computales.py:106:80: E360 line too long (83 > 79 characters)
computales.py:106:80: E360 line too long (80 > 79 characters)
computales.py:106:22: E331 missing whitespace after ':'
computales.py:106:24: W291 trailing whitespace
computales.py:106:24: W291 trailing whitespace
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> flake8 computales.py
computales.py:4:1: W291 trailing whitespace
computales.py:14:1: E360 line too long (91 > 79 characters)
computales.py:14:25: E733 test for membership should be 'not in'
computales.py:14:40: E360 line too long (91 > 79 characters)
computales.py:16:40: E360 line too long (91 > 79 characters)
computales.py:16:80: E360 line too long (81 > 79 characters)
computales.py:16:80: E360 line too long (91 > 79 characters)
computales.py:106:80: E360 line too long (83 > 79 characters)
computales.py:106:80: E360 line too long (80 > 79 characters)
computales.py:106:22: E331 missing whitespace after ':'
computales.py:106:24: W291 trailing whitespace
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> flake8 computales.py
computales.py:62:21: E128 continuation line under-indented for visual indent
computales.py:62:21: E733 test for membership should be 'not in'
computales.py:62:51: W291 trailing whitespace
computales.py:111:15: E501 f-string is missing placeholders
computales.py:111:22: E331 missing whitespace after ':'
computales.py:111:22: E331 missing whitespace after ':'
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> flake8 computales.py
computales.py:62:21: E128 continuation line under-indented for visual indent
computales.py:62:21: E733 test for membership should be 'not in'
computales.py:62:51: W291 trailing whitespace
computales.py:111:15: E501 f-string is missing placeholders
computales.py:111:22: E331 missing whitespace after ':'
computales.py:62:25: E733 test for membership should be 'not in'
computales.py:62:51: W291 trailing whitespace
computales.py:111:15: E501 f-string is missing placeholders
computales.py:111:22: E331 missing whitespace after ':'
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> flake8 computales.py
computales.py:62:25: E733 test for membership should be 'not in'
computales.py:62:51: W291 trailing whitespace
computales.py:111:15: E501 f-string is missing placeholders
computales.py:111:22: E331 missing whitespace after ':'
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> flake8 computales.py
computales.py:71:1: W291 blank line contains whitespace
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> flake8 computales.py
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4>
```

4. Pruebas unitarias pasadas correctamente:



```
1 import unittest
2 from computales import Computales
3
4 class TestComputales(unittest.TestCase):
5
6     def setUp(self):
7         self.test_cases = [
8             {'catalogue_file': 'TC1/TC1.ProductList.json', 'sales_file': 'TC1/TC1.Sales.json', 'expected_total': 2483.86},
9             {'catalogue_file': 'TC2/TC1.ProductList.json', 'sales_file': 'TC2/TC2.Sales.json', 'expected_total': 150235.37},
10            {'catalogue_file': 'TC3/TC1.ProductList.json', 'sales_file': 'TC3/TC3.Sales.json', 'expected_total': 2483.86}
11        ]
12
13     def test_TC1(self):
14         catalogue_data = self.test_cases[0]['catalogue_file']
15         sales_data = self.test_cases[0]['sales_file']
16
17         sales_processor = Computales(catalogue_data, sales_data)
18         sales_processor.process_sales()
19         expected_total_cost = self.test_cases[0]['expected_total_cost']
20         self.assertEqual(sales_processor.total_cost, expected_total_cost)
21
22     def test_TC2(self):
23         catalogue_data = self.test_cases[1]['catalogue_file']
24         sales_data = self.test_cases[1]['sales_file']
25
26         sales_processor = Computales(catalogue_data, sales_data)
27         sales_processor.process_sales()
28         expected_total_cost = self.test_cases[1]['expected_total_cost']
29         self.assertEqual(sales_processor.total_cost, expected_total_cost)
30
31     def test_TC3(self):
32         catalogue_data = self.test_cases[2]['catalogue_file']
33         sales_data = self.test_cases[2]['sales_file']
34
35         sales_processor = Computales(catalogue_data, sales_data)
36         sales_processor.process_sales()
37         expected_total_cost = self.test_cases[2]['expected_total_cost']
38         self.assertEqual(sales_processor.total_cost, expected_total_cost)
39
```

```
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> python -m unittest test_compute_sales
4
ERROR: test_TC3 (test_compute_sales.TestComputales.test_TC3)
Traceback (most recent call last):
  File "C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4\test_compute_sales.py", line 14, in test_TC3
    catalogue_data = self.load_json_file(self.test_cases[0]['catalogue_file'])
AttributeError: 'test_compute_sales.TestComputales' object has no attribute 'load_json_file'

Run 1 test in 0.006s

FAILED (errors=1)
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> python -m unittest test_compute_sales
----- Sales Results -----
Total sales cost: $2483.86
Time elapsed: 0.0015430408 seconds
f

FAIL: test_TC1 (test_compute_sales.TestComputales.test_TC1)
Traceback (most recent call last):
  File "C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4\test_compute_sales.py", line 20, in test_TC1
    self.assertEqual(sales_processor.total_cost, expected_total_cost)
AssertionError: 2483.86 != 2401.86

Run 1 test in 0.002s

FAILED (failures=1)
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4> python -m unittest test_compute_sales
----- Sales Results -----
Total sales cost: $2483.86
Time elapsed: 0.000506395 seconds

>Error: Product "frijoles" not found in the price catalogue.
Error: Product "frijoles" not found in the price catalogue.
Sales Results
Total sales cost: $150235.37
Time elapsed: 0.000999275 seconds

Run 2 tests in 0.006s
OK
PS C:\Users\erick\Documents\VA\Calidad y Pruebas de Software\Tarea 4>
```