PIYUSH KUMAR MISHRA

230957212

ROLL N0:70

WEEK 8:

Exercise 1:

File Handling in Python with Error and Exception Handling

You are tasked with building a simple data processing system for an e-commerce company that handles product inventories and customer reviews. The company uses two types of files to store data:

1. CSV Files: The product inventory is stored in a CSV file with the following columns: Product IL), Product Name, Category, Price, and Stock Quantity.

2, JSON Files: Customer reviews for each product are stored in separate JSON files. Each review file contains a list of reviews for a product, where each review has the following fields: Review IL), User ID, Rating, Comment, and Date.

Your system needs to process these files to provide insights and updates on product availability, pricing trends, and customer feedback. Additionally, the system should allow adding new products and updating product information in the CSV file, while handling errors and exceptions gracefully.

System Requirements:

1. Product Management (CSV File):

• Add New Products: Write a function to add a new product to the inventory CSV file.

• Update Stock and Price: Implement a function to update the stock quantity and price of an existing product in the CSV file.

Check Product Availability: Write a function that reads the CSV file to check if a product is in stock and, if so, how many units are available.

2. Customer Review Management (JSON Files):

Add Customer Review: Implement a function that adds a new customer review to the corresponding JSON file for a product.

Average Rating Calculation: Write a function that reads a product's JSON review file and calculates its average rating.

Review Search: Implement a search function that reads the JSON file and allows users to search for reviews containing specific keywords in the comment section.

3. Data Analysis:

• Top Rated Products: Create a function that reads all product review JSON files, calculates the average rating for each product, and lists the top 5 highest-rated products.

• Out of Stock Products: Write a function that reads the CSV inventory file and lists all products that are out of stock.

• Price Trends: Implement a function to read the CSV file and identify products whose prices have increased or decreased in the past month.

4. Error and Exception Handling:

• Implement error handling for situations such as:

File Not Found: When the CSV or JSON file is missing.

Invalid Data Format: If the data in the files does not match the expected structure.

File Corruption: When a file contains corrupted data.

Read/Write Permissions: When the system does not have permission to read or write to a file.

```
import csv
import json
import os
```

```
def add_product(file_path):
    product_id = input("Enter Product ID: ")
    product_name = input("Enter Product Name: ")
    category = input("Enter Category: ")
    price = float(input("Enter Price: "))
    stock_quantity = int(input("Enter Stock Quantity: "))

    try:
        with open(file_path, mode='a', newline='') as file:
            writer = csv.writer(file)
            writer.writerow([product_id, product_name, category, price, stock_quantity])
            print("Product added successfully.")
    except PermissionError:
        print("Error: Permission denied when trying to write to the file.")
    except Exception as e:
        print(f"An unexpected error occurred: {e}")

def update_product(file_path):
    product_id = input("Enter Product ID to update: ")
    new_price = float(input("Enter New Price: "))
    new_stock_quantity = int(input("Enter New Stock Quantity: "))

    try:
        updated_rows = []
        with open(file_path, mode='r') as file:
            reader = csv.reader(file)
```

```
            for row in reader:
                if row[0] == product_id:
                    row[3] = str(new_price)
                    row[4] = str(new_stock_quantity)
                updated_rows.append(row)

        with open(file_path, mode='w', newline='') as file:
            writer = csv.writer(file)
            writer.writerows(updated_rows)
            print("Product updated successfully.")
    except FileNotFoundError:
        print("Error: The CSV file was not found.")
    except Exception as e:
        print(f"An unexpected error occurred: {e}")

def check_availability(file_path):
    product_id = input("Enter Product ID to check availability: ")

    try:
        with open(file_path, mode='r') as file:
            reader = csv.reader(file)
            for row in reader:
                if row[0] == product_id:
                    return f"Available stock: {row[4]}"
        return "Product not found."
    except FileNotFoundError:
        print("Error: The CSV file was not found.")
```

```
def add_review():
    product_id = input("Enter Product ID for the review: ")
    review_id = input("Enter Review ID: ")
    user_id = input("Enter User ID: ")
    rating = int(input("Enter Rating (1-5): "))
    comment = input("Enter Comment: ")
    date = input("Enter Date (YYYY-MM-DD): ")

    review = {
        "Review ID": review_id,
        "User ID": user_id,
        "Rating": rating,
        "Comment": comment,
        "Date": date
    }

    file_name = f"{product_id}_reviews.json"

    try:
        if not os.path.exists(file_name):
            with open(file_name, 'w') as file:
                json.dump([], file)

        with open(file_name, 'r+') as file:
            reviews = json.load(file)
            reviews.append(review)
```

```
            file.seek(0)
            json.dump(reviews, file, indent=4)
            print("Review added successfully.")
    except Exception as e:
        print(f"An error occurred while adding the review: {e}")

def average_rating(product_id):
    file_name = f"{product_id}_reviews.json"

    try:
        with open(file_name) as file:
            reviews = json.load(file)
            total_rating = sum(review['Rating'] for review in reviews)
            average = total_rating / len(reviews) if reviews else 0
            return f"Average Rating: {average:.2f}"
    except FileNotFoundError:
        return "Review file not found."
    except json.JSONDecodeError:
        return "Error: Invalid JSON format."

def search_reviews():
    product_id = input("Enter Product ID to search reviews: ")
    keyword = input("Enter keyword to search in comments: ")

    file_name = f"{product_id}_reviews.json"

    try:
```

```
        with open(file_name) as file:
            reviews = json.load(file)
            matching_reviews = [review for review in reviews if keyword.lower() in
review['Comment'].lower()]
            return matching_reviews if matching_reviews else "No matching reviews found."
    except FileNotFoundError:
        return "Review file not found."

def top_rated_products(product_ids):
    ratings = {}

    for product_id in product_ids:
        avg_rating_result = average_rating(product_id)

        if isinstance(avg_rating_result, str) and "Average Rating" in avg_rating_result:
            ratings[product_id] = float(avg_rating_result.split(": ")[1])

    top_products = sorted(ratings.items(), key=lambda x: x[1], reverse=True)[:5]

    return top_products

def out_of_stock_products(file_path):
    out_of_stock = []

    try:
        with open(file_path) as file:
            reader = csv.reader(file)
            for row in reader:
```

```
            if int(row[4]) == 0:
                out_of_stock.append(row[1])

        return out_of_stock if out_of_stock else ["No products are out of stock."]

    except FileNotFoundError:
        return "Inventory file not found."

def price_trends(current_file_path, previous_file_path):
    current_prices = {}
    previous_prices = {}

    try:
        with open(current_file_path) as current_file:
            reader = csv.reader(current_file)
            for row in reader:
                current_prices[row[0]] = float(row[3])

        with open(previous_file_path) as previous_file:
            reader = csv.reader(previous_file)
            for row in reader:
                previous_prices[row[0]] = float(row[3])

        trends = {}
        for product_id in current_prices.keys():
            if product_id in previous_prices:
                change = current_prices[product_id] - previous_prices[product_id]
```

```
                trends[product_id] = change

        return trends
    except FileNotFoundError as e:
        return f"File not found: {e.filename}"

def main():
    inventory_file_path = 'inventory.csv'

    while True:
        print("\nE-commerce Data Processing System")
        print("1. Add Product")
        print("2. Update Product")
        print("3. Check Product Availability")
        print("4. Add Customer Review")
        print("5. Calculate Average Rating")
        print("6. Search Reviews")
        print("7. List Out of Stock Products")
        print("8. Show Top Rated Products")

        choice = input("\nSelect an option (or 'q' to quit): ")

        if choice == '1':
            add_product(inventory_file_path)
        elif choice == '2':
            update_product(inventory_file_path)
        elif choice == '3':
```

```
    print(check_availability(inventory_file_path))
elif choice == '4':
    add_review()
elif choice == '5':
    product_id = input("Enter Product ID to calculate average rating: ")
    print(average_rating(product_id))
elif choice == '6':
    results = search_reviews()
    if isinstance(results, list):
        for review in results:
            print(review)
    else:
        print(results)
elif choice == '7':
    out_of_stock_list = out_of_stock_products(inventory_file_path)
    for item in out_of_stock_list:
        print(item)
elif choice == '8':
    example_product_ids = ['P001', 'P002', 'P003', 'P004', 'P005']
    top_products_list = top_rated_products(example_product_ids)
    for product in top_products_list:
        print(product)
elif choice.lower() == 'q':
    break
else:
    print("Invalid option. Please try again.")
```

```
if __name__ == "__main__":
    main()
```

OUPUT:

```
E-commerce Data Processing System
1. Add Product
2. Update Product
3. Check Product Availability
4. Add Customer Review
5. Calculate Average Rating
6. Search Reviews
7. List Out of Stock Products
8. Show Top Rated Products

Select an option (or 'q' to quit):  1
Enter Product ID:  123
Enter Product Name:  XYZ
Enter Category:  GROCERY
Enter Price:  1000
Enter Stock Quantity:  9
Product added successfully.

E-commerce Data Processing System
1. Add Product
2. Update Product
3. Check Product Availability
4. Add Customer Review
5. Calculate Average Rating
6. Search Reviews
7. List Out of Stock Products
8. Show Top Rated Products

Select an option (or 'q' to quit):  q
```