Optimizing Inventory Management with Bubble Sort

Sumeya Sayd
06/26/2024

Adapting Bubble Sort for inventory management in retail stores

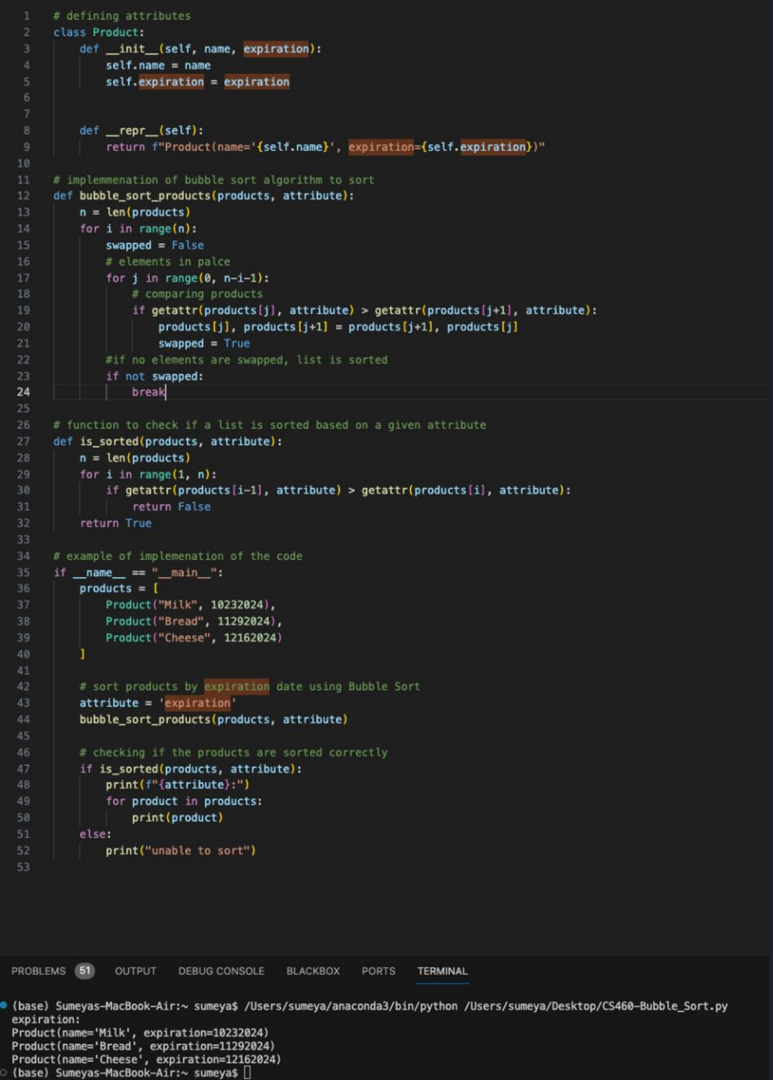
- Bubble Sort is an algorithm known for simplicity in sorting data, which I will now apply to enhance retail operations.
- Retail faces challenges like fluctuating product demand and managing items that can go bad
- Key Challenges:
    - Variability in demand
    - Shelf life and expiration dates
    - Optimal stock maintenance for customer satisfaction and waste reduction.

Adapting Bubble Sort for Retail

- **Algorithm Adaptation:**
  - Modify Bubble Sort to prioritize sorting by expiration dates.
  -
- **Implementation:**
  - Custom sorting criteria tailored to retail needs
  - Integration into existing inventory systems for seamless operations.

Sample code in next slide

Explanation of Code Implementation



```python
# defining attributes
class Product:
    def __init__(self, name, expiration):
        self.name = name
        self.expiration = expiration


    def __repr__(self):
        return f"Product(name='{self.name}', expiration={self.expiration})"

# implemmenation of bubble sort algorithm to sort
def bubble_sort_products(products, attribute):
    n = len(products)
    for i in range(n):
        swapped = False
        # elements in palce
        for j in range(0, n-i-1):
            # comparing products
            if getattr(products[j], attribute) > getattr(products[j+1], attribute):
                products[j], products[j+1] = products[j+1], products[j]
                swapped = True
        #if no elements are swapped, list is sorted
        if not swapped:
            break

# function to check if a list is sorted based on a given attribute
def is_sorted(products, attribute):
    n = len(products)
    for i in range(1, n):
        if getattr(products[i-1], attribute) > getattr(products[i], attribute):
            return False
    return True

# example of implemenation of the code
if __name__ == "__main__":
    products = [
        Product("Milk", 10232024),
        Product("Bread", 11292024),
        Product("Cheese", 12162024)
    ]

    # sort products by expiration date using Bubble Sort
    attribute = 'expiration'
    bubble_sort_products(products, attribute)

    # checking if the products are sorted correctly
    if is_sorted(products, attribute):
        print(f"{attribute}:")
        for product in products:
            print(product)
    else:
        print("unable to sort")
```

```
(base) Sumeyas-MacBook-Air:~ sumeya$ /Users/sumeya/anaconda3/bin/python /Users/sumeya/Desktop/CS460-Bubble_Sort.py
expiration:
Product(name='Milk', expiration=10232024)
Product(name='Bread', expiration=11292024)
Product(name='Cheese', expiration=12162024)
(base) Sumeyas-MacBook-Air:~ sumeya$ 
```

- **Bubble Sort Algorithm:**
  - This algorithm iterates over the product list products, comparing products based on the expiration date
  - Products are swapped if they are out of order, ensuring the list is sorted in ascending order by the end of each pass.
  - The algorithm optimizes inventory management by organizing products efficiently based on their expiration date

- **Sorting Validation:**
  - is_sorted function checks if the list of products is sorted correctly.
  - It checks pairwise comparisons based on the given attribute to confirm the order.
  - This ensures products are arranged appropriately, crucial for maintaining inventory accuracy and operational efficiency.

- **Example Implementation:**
  - In the example provided:
    - Products are initialized with names and expiration dates.
    - The products are sorted by their expiration dates using the implemented Bubble Sort algorithm.
    - The sorted products are then displayed to confirm the sorting accuracy.
    - This demonstrates the practical application of Bubble Sort in sorting perishable goods effectively within retail inventory management.

Conclusion

- **Summary of Benefits:**
  - Bubble Sort enhances inventory management by prioritizing the expiration date of each product.
  - Successfully implemented in retail to optimize operations and customer satisfaction.
- **Future Enhancements:**
  - Scale algorithm for larger inventories and integrate real-time data analytics.
  - Explore applications beyond retail for similar challenges