Due Date: 09.06.2024 23:59

CENG112 – Data Structures
Homework #3

In this homework, you are expected to implement a product inventory management system using Java that sorts products by category first and then by name within each category.:

- Generics
- Search
- Trees
- Binary Search Tree
- CSV file I/O

The focus is on searching, sorted lists, and binary search tree implementation. Inputs will be provided through a CSV file. Format for the CSV is given below:

add,Category,Product,Price

search,Product

list

Example:

add,Fruit,Apple,20

add,Vegetable,Carrot,25

list

add,Fruit,Orange,30

search,Apple

search,Soap

add,Dairy, Cheese,100

list

.

.

.

Implementation

- Product Class:
  - Create a Product class with attributes name, category, and price.

- Node Class:
  - Create a generic Node<T> class to represent nodes in the tree.

- Tree Class:

- Implement a generic Tree<T> class that uses a binary search tree to manage the products.
- Include methods to add a node, search for a node by custom criteria, and list all nodes in sorted order.

- Inventory Class:
  - Implement an Inventory class that uses the Tree<Product> to manage the products.

- FileIO Class:
  - Create a FileIO class to handle reading products and commands from CSV files.

- Main Class:
  - Create a Main class that reads commands from a CSV file to interact with the product inventory.

Example Output:

Adding:

Product: Apple, Category: Fruit, Price: 20₺

Adding:

Product: Carrot, Category: Vegetable, Price: 25₺

Listing:

Apple (Fruit): 20₺

Carrot (Vegetable): 25₺

Adding:

Product: Orange, Category: Fruit, Price: 30₺

Searching:

Found: Apple (Fruit): 20₺

Searching:

Not Found: Soap

Adding:

Product: Cheese, Category: Dairy, Price: 100₺

List:

Cheese (Dairy): 100₺

Apple (Fruit): 20₺

Orange (Fruit): 30₺

Carrot (Vegetable): 25₺

.

.

Important Notes:

1. Do NOT request inputs in your app. Printing the results of the queries will be enough.
2. You can use standard **java.io** packages to read files. Do NOT use other 3rd party libraries.
3. You should use **relative** paths (e.g. Files/sample.csv) instead of **absolute** paths (e.g. C:\\user\\eclipse-workspace\\MyProject\\Files\\sample.csv). Please be sure of it, otherwise there will be **no output** of your application and you certainly will **lose points**.
4. To support **Turkish characters,** you may need to change your project's text file encoding to UTF8: Right click on your project (in package explorer) → Properties → Text file encoding → Other → UTF8 → Apply.
5. You are expected to write clean, readable, and tester-friendly code. Please try to maximize reusability and prevent from redundancy in your methods.

Assignment Rules:

1. In this lecture's homework, there are no cheating allowed. If any cheating has been detected, they will be graded as 0 and there will be no further discussion on this.
2. You are expected to submit your homework in groups. Therefore, only one of you will be sufficient to submit your homework.
3. Make sure you export your homework as an Eclipse project. You can use other IDEs as well, however, you must test if it **can be executed** in Eclipse.
4. Your exported Java Project should have the following naming format with your assigned group ID as the given below:

   G05_CENG112_HW3

   Also the zip folder that your project in should have the same name

   G05_CENG112_HW3.zip
5. Please beware that if you do not follow the assignment rules for exporting and naming conventions, you will lose points.