Tutorial-6

You have been approached by a startup ecommerce company **Amacon** to create a software application that could be used to sell their products. Your software implementation must use object oriented programming approach. You must also use Java Collection Framework and exception handling techniques where you define and throw your own specialized exceptions. Amacon software application involves three entities: **database**, **administrator**, and **customer**.

The **database** would be updated only by the **administrator** and it should support the following operations:

1. The database should have the capability to **insert** a new category or a new product. Each product in the database should have a price and the number of units available. Each product is listed under exact one category. For example, in the path “electronics > smartphone>Oneplus", “Oneplus” is a product listed under subcategory “smartphone” and that in turn is under the top level category “electronics”. This method should be called with two parameters: first is the path to the subcategory and second parameter is the product to be added in that subcategory. For the same example mentioned above, call to insert will be as: **insert("electronics > smartphone", "Oneplus")**. If that subcategory does not exist in the database then this function should create the above subcategory along with the product mentioned. If the database already has that product under that category then program should raise a custom exception. While adding this product in the database, insert should take input from the administrator about the properties of that product.

2. The database should have the capability to **delete** a sub-category/ product. Example, this method could be called in either of following two ways: **delete("electronic > smartphone>Oneplus")** or **delete("electronic > smartphone").** The first one just delete the product Oneplus but the second one removes all the subcategory/products under “smartphone” subcategory. If invalid path is specified in delete API then program should raise a custom exception.

3. The database should have the capability to **search** for a product and is called as **search(“Oneplus”)**. If the product does not exist, the program should raise a custom exception. If the product exists, then the search method will print the path to the product as well as details of the product. This method should also return the reference to the product.

4. The database should have the capability to **modify** the properties of a product, i.e change its price or the number of units in stock. It can internally call **search** API described above to get the reference to the product for updating the properties of that product. It can take input from administrator for the properties to be updated.

5. The database should be able to handle the **sale** of a product. This API would be called by the customer after he has added some products in his cart. This API accepts 3 parameters as follows: **sale(product, quantity, remaining_funds_with_customer)**. If that much of stock is unavailable or if funds are insufficient to buy this product the

program should raise appropriate custom exceptions.

Customer

A customer has to explicitly add funds. He is provided with a shopping cart that can hold multiple products of his choice. This cart should support the following operations:

1. The cart should have the capability to **add a product** and store it. This should make use of **search** operation on the database. While adding a product to the cart customer should add the quantity to purchase as well.

2. The cart should have the capability to **check-out**, i.e buy **all** products in the cart. This should make use of **sale** operation on the database.

You have to implement a menu-driven system that supports the following queries. A user can act as an administrator and a customer (both only one per program invocation), which must be prompted as input before making the following queries:

1. Administrator
    a. Insert product/category
    b. Delete product/category
    c. Search product
    d. Modify product
    e. Exit as Administrator
2. Customer
    a. Add funds (assume all integer operations)
    b. Add product to the cart
    c. Check-out cart
    d. Exit as Customer

The program should loop with above two modes unless explicitly terminated. Before termination the program should print the total revenue generated by Amacon.