| | | |
|---|---|---|
| **OOP Lab**<br>Exp6: Multithreading | Name | Shubham Goel |
| | Roll No | 2019130015 |
| | Batch | A |
| | Date | 22 Nov. 21 |
| | Branch | COMPS |

**Aim:** Implement Multithreading

## Theory

Thread class is the main class on which Java's Multithreading system is based. Thread class, along with its companion interface Runnable will be used to create and run threads for utilizing Multithreading feature of Java.

It provides constructors and methods to support multithreading. It extends object class and implements Runnable interface.

## Code:

## Main.java

Multithreading

```
import java.util.*;
import java.lang.*;

//custom exception
class InvalidPrice extends Exception {
    public InvalidPrice(String str) {
        super(str);
    }
}

// Base class
class User {
    String name;
    int age;
```

```
    void printUserStamp() {
        System.out.println("");
        System.out.println("Inside the user class");
        System.out.println("");
    }

    User() {
        this.name = "a_user";
        this.age = 100;
    }

    void printUserInfo() {
    };

    void inputNameAge() {
        // printUserStamp();
        Scanner scanner_obj = new Scanner(System.in); // Create a
Scanner object
        System.out.println("Enter the name of the user");
        this.name = scanner_obj.nextLine();
        System.out.println("Enter the age");
        this.age = scanner_obj.nextInt();
    }

    void printUserDetails() {
        System.out.println(this.name);
        System.out.println(this.age);
    }
}

class Seller extends User {
    String GSTno;
    int rating;
    Product sellerProducts[] = new Product[100]; // aggregation
    int lastFreeIndex = 0;

    Seller() {
        this.GSTno = "123";
        this.rating = 3;
    }

    void addProduct(String name, int price) {
```

```
        Product addProduct = new Product(name, price);
        this.sellerProducts[this.lastFreeIndex++] = addProduct;
    }

    void printSellerProducts() {
        for (int i = 0; i < this.lastFreeIndex; i++) {
            System.out.println("Index:" + i);
            System.out.println("price = " +
this.sellerProducts[i].price);
            System.out.println("name = " +
this.sellerProducts[i].name);
            System.out.println();
        }
    }

    void printSellerInfo() {
        printUserDetails();
        System.out.println(this.GSTno);
        System.out.println(this.rating);
    }

    void printSellerStamp() {
        System.out.println("");
        System.out.println("Inside the Seller class");
        System.out.println("");
    }

    void inputSellerDetails() {
        // printSellerStamp();

        inputNameAge();
        Scanner scan_obj = new Scanner(System.in); // Create a
Scanner object
        System.out.println("Enter GSTno for the user");
        this.GSTno = scan_obj.nextLine();
        System.out.println("Enter the rating for the seller");
        this.rating = scan_obj.nextInt();
        // scan_obj.close();
    }
}

class Cart implements Runnable {
```

```
    Product cartArr[] = new Product[100];
    int firstFreeIndex;

    void printCartDetails() {

        for (int i = 0; i < firstFreeIndex; i++) {
            System.out.println(this.cartArr[i].name);
        }
    }

    void addProductToCart(Product productToAdd) {
        this.cartArr[this.firstFreeIndex++] = productToAdd;
    }

    public void run() {
        int max_product = Product.lastProductLoc;
        for (int i = 0; i < max_product; i++) {
            System.out.println("Adding Product to first customer
cart");
            this.addProductToCart(Product.allProducts[i]);
            try {
                Thread.sleep(500);
            } catch (Exception e) {
            }
        }
    }
}

class Buyer extends User implements Runnable {
    String address;
    String creditCardID;
    Cart cart;

    Buyer(String name) {
        this.cart = new Cart();
        this.name = name;
    }

    void printBuyerInfo() {
        printUserDetails();
        System.out.println(this.address);
        System.out.println(this.creditCardID);
```

```
    }

    void printBuyerStamp() {
        System.out.println("");
        System.out.println("Inside the buyer class");
        System.out.println("");
    }

    void inputBuyerDetails() {
        // printBuyerStamp();

        inputNameAge();
        Scanner scan_obj = new Scanner(System.in); // Create a
Scanner object
        System.out.println("Enter address for the user");
        this.address = scan_obj.nextLine();
        System.out.println("Enter the credit card id");
        this.creditCardID = scan_obj.nextLine();
        // scan_obj.close();
    }

    public void run() {

        for (int i = 0; i < this.cart.firstFreeIndex; i++) {
            System.out.println("Buyer " + this.name + " " + "bought
" + this.cart.cartArr[i].name);
            try {
                Thread.sleep(500);
            } catch (Exception e) {
            }
        }
    }

}

// composition

class Product {
    String name;
    int price;
    int quantity;
    static Product allProducts[] = new Product[400];
```

```
    static int lastProductLoc = 0;

    static void addToAllProducts(Product toAdd) {
        allProducts[lastProductLoc++] = toAdd;

    }

    Product(String name, int price) {
        this.name = name;
        this.price = price;
        this.quantity = 10;

        addToAllProducts(this);
    }

    static void printAllProducts() {
        for (int i = 0; i < lastProductLoc; i++) {
            System.out.println("Index:" + i);
            System.out.println("price = " + allProducts[i].price);
            System.out.println("name = " + allProducts[i].name);
            System.out.println();
        }
    }

    void printProductDetails() {
        System.out.println("price = " + this.price);
        System.out.println("name = " + this.name);
        System.out.println();
    }

}

public class Main {
    // validate the price
    static void validatePrice(int price) throws InvalidPrice {
        if (price < 0) {
            throw new InvalidPrice("The price should be positive");
        } else {
            System.out.println("The price is valid");
        }
    }
```

```
    public static void main(String args[]) {
        Buyer a = new Buyer("first");
        Buyer b = new Buyer("second");
        Product pa = new Product("p1", 200);
        Product pb = new Product("p2", 200);
        Product pc = new Product("p3", 200);
        Product pd = new Product("p4", 200);
        Product pe = new Product("p5", 200);

        // add products to 1st customer in one thread
        Thread insert_all_product_to_buyer_one = new Thread(a.cart);
        insert_all_product_to_buyer_one.start();

        Thread aThread = new Thread(a);
        Thread bThread = new Thread(b);

        b.cart.addProductToCart(pa);
        b.cart.addProductToCart(pd);
        b.cart.addProductToCart(pe);

        // buy the cart item with multi-threading
        bThread.start();
        while (insert_all_product_to_buyer_one.isAlive() == true) {
// wait for cart to complete
            try {
                Thread.sleep(500);
            } catch (Exception e) {
            }
        }
        aThread.start();

        System.out.println();

    }
}
```

## Output:

```
Adding Product to first customer cart
Buyer second bought p1
Adding Product to first customer cart
Buyer second bought p4
Adding Product to first customer cart
Buyer second bought p5
Adding Product to first customer cart
Adding Product to first customer cart

Buyer first bought p1
Buyer first bought p2
Buyer first bought p3
Buyer first bought p4
Buyer first bought p5
```

## Conclusion

I have learnt and implemented multi-threading .

Reference:

https://www.javatpoint.com/access-modifiers