CSE-214 Online - 2 (C1)
Structural Design Pattern

In your locality, there's a cozy coffee place named **CoffeeTong** that serves two types of coffee: **Americano** and **Cappuccino**.

- **Americano**: A type of black coffee prepared by adding extra grinded coffee beans to regular black coffee.
- **Cappuccino**: A type of milk coffee prepared by adding cinnamon powder to regular milk coffee.

Coffee Preparation Details:

- **Black Coffee**: Prepared using water and grinded coffee beans.
- **Milk Coffee**: Prepared using milk and grinded coffee beans.

Cost Breakdown:

- CoffeTong serves coffee in their handmade fancy mugs, each of which costs **100 taka**.
- Grinded coffee beans (used in both types) add **30 taka** per cup.
- Milk (used in milk coffee only) adds **50 taka** per cup.
- Extra ingredients:
    - **Americano**: Additional grinded coffee beans, adding **30 taka**.
    - **Cappuccino**: Cinnamon powder, adding **50 taka**.

Requirements:

Implement a program using an appropriate design pattern to manage different coffee types and their variations, ensuring that the coffee preparation process is easily extendable if new coffee types are added in the future. The base Coffee class and the driver class is provided for your convenience.

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;


interface Coffee {
    String getIngredients();
    int getCost();
```

```
}

// Order class to handle multiple coffee orders
class Order {
    private List<Coffee> coffees = new ArrayList<>();

    public void addCoffee(Coffee coffee) {
        coffees.add(coffee);
    }

    public void printOrderDetails() {
        int totalCost = 0;
        int coffeeCount = 1;

        for (Coffee coffee : coffees) {
            System.out.println("Coffee " + coffeeCount + ":");
            System.out.println("Ingredients: " + coffee.getIngredients());
            System.out.println("Cost: " + coffee.getCost() + " taka");
            System.out.println();
            totalCost += coffee.getCost();
            coffeeCount++;
        }

        System.out.println("Total Cost for Order: " + totalCost + " taka");
    }
}

// Main Class

public class CoffeeTong {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Order order = new Order();

        while (true) {
            System.out.println("Select coffee type (1: Americano, 2: Espresso, 3: Cappuccino, 4:
Mocha, 0: Finish): ");
            int choice = scanner.nextInt();
            if (choice == 0) break;

            Coffee coffee;
            switch (choice) {
                case 1:
                    coffee = new Americano(new BasicBlackCoffee());
                    break;
                case 3:
                    coffee = new Cappuccino(new BasicMilkCoffee());
                    break;
                default:
                    System.out.println("Invalid choice.");
```

```
                continue;
            }
            order.addCoffee(coffee);
        }

        order.printOrderDetails();
    }
}
```