```
Void main() {
  // Sample list of item prices
  List<double> itemPrices = [5.99, 15.49, 23.89, 3.50, 9.99, 50.00];

  // Calculate the total price without any discount or tax
  Double totalPrice = calculateTotal(itemPrices);
  Print('Total price without discount or tax: \$${totalPrice.toStringAsFixed(2)}');

  // Apply discount using a higher-order function
  Double discountPercentage = 10; // Discount percentage (e.g., 10%)
  Double discountedPrice = applyDiscount(itemPrices, (price) {
    Return price * (1 – discountPercentage / 100);
  });
  Print('Total after applying discount: \$${discountedPrice.toStringAsFixed(2)}');

  // Calculate total price with tax (optional parameter for tax)
  Double taxRate = 0.08; // 8% tax
  Double totalWithTax = calculateTotal(itemPrices, tax: taxRate);
  Print('Total price with tax: \$${totalWithTax.toStringAsFixed(2)}');

  // Filter out items below a certain price using an anonymous function
  Double priceThreshold = 10.0;
  Var filteredItems = itemPrices.where((price) => price >= priceThreshold).toList();
  Print('Items priced above \$${priceThreshold.toStringAsFixed(2)}: $filteredItems');

  // Apply the factorial-based discount using recursion
```

```
  Double specialDiscountPrice = applyFactorialDiscount(discountedPrice,
itemPrices.length);

  Print('Price after applying special factorial discount:
\$${specialDiscountPrice.toStringAsFixed(2)}');

}


// Function to calculate the total price

Double calculateTotal(List<double> prices, {double tax = 0.0}) {

  Double total = prices.fold(0, (sum, price) => sum + price);

  Return total + (total * tax); // Add tax if provided

}


// Higher-order function to apply a discount

Double applyDiscount(List<double> prices, double Function(double) discountFunction) {

  List<double> discountedPrices = prices.map(discountFunction).toList();

  Return discountedPrices.fold(0, (sum, price) => sum + price); // Sum the discounted
prices

}


// Recursive function to calculate factorial and apply special discount

Double applyFactorialDiscount(double price, int numberOfItems) {

  // Calculate the factorial of the number of items

  Int factorialValue = factorial(numberOfItems);

  Double discountPercentage = factorialValue / 100.0; // Convert factorial to a percentage

  Return price * (1 – discountPercentage); // Apply the factorial-based discount

}
```

```
// Helper function to calculate the factorial of a number
Int factorial(int n) {
  If (n == 1 || n == 0) {
    Return 1;
  } else {
    Return n * factorial(n – 1);
  }
}
```