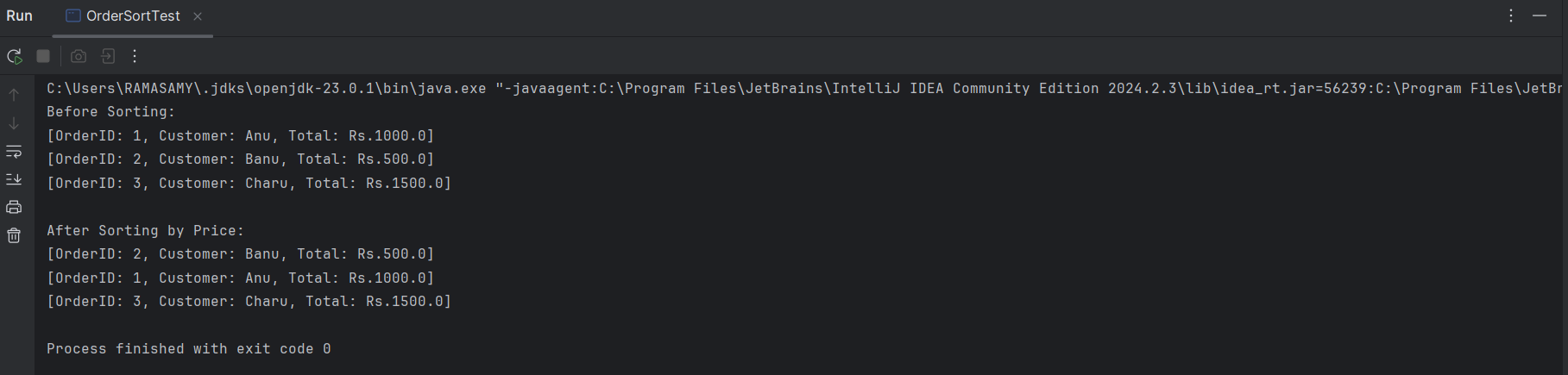
**Exercise 3: Sorting Customer Orders**

**CODE:**

**OrderSortTest.java**

class Order {  
 int orderId;  
 String customerName;  
 double totalPrice;  
  
 public Order(int orderId, String customerName, double totalPrice) {  
 this.orderId = orderId;  
 this.customerName = customerName;  
 this.totalPrice = totalPrice;  
 }  
  
 public String toString() {  
 return "[OrderID: " + orderId + ", Customer: " + customerName + ", Total: Rs." + totalPrice + "]";  
 }  
}  
  
class OrderSorter {  
 public static void bubbleSort(Order[] orders) {  
 int n = orders.length;  
 for (int i = 0; i < n - 1; i++) {  
 for (int j = 0; j < n - i - 1; j++) {  
 if (orders[j].totalPrice > orders[j + 1].totalPrice) {  
 Order temp = orders[j];  
 orders[j] = orders[j + 1];  
 orders[j + 1] = temp;  
 }  
 }  
 }  
 }  
  
 public static void quickSort(Order[] orders, int low, int high) {  
 if (low < high) {  
 int pi = *partition*(orders, low, high);  
 *quickSort*(orders, low, pi - 1);  
 *quickSort*(orders, pi + 1, high);  
 }  
 }  
  
 private static int partition(Order[] orders, int low, int high) {  
 double pivot = orders[high].totalPrice;  
 int i = low - 1;  
 for (int j = low; j < high; j++) {  
 if (orders[j].totalPrice < pivot) {  
 i++;  
 Order temp = orders[i];  
 orders[i] = orders[j];  
 orders[j] = temp;  
 }  
 }  
 Order temp = orders[i + 1];  
 orders[i + 1] = orders[high];  
 orders[high] = temp;  
 return i + 1;  
 }  
}  
  
class OrderSortTest {  
 public static void main(String[] args) {  
 Order[] orders = {  
 new Order(1, "Anu", 1000),  
 new Order(2, "Banu", 500),  
 new Order(3, "Charu", 1500)  
 };  
  
 System.*out*.println("Before Sorting:");  
 for (Order o : orders) System.*out*.println(o);  
  
 // OrderSorter.bubbleSort(orders);  
 OrderSorter.*quickSort*(orders, 0, orders.length - 1);  
  
 System.*out*.println("\nAfter Sorting by Price:");  
 for (Order o : orders) System.*out*.println(o);  
 }  
}

**OUTPUT:**

****