**Module 2**

**Exercise 3: Sorting Customer Orders**

**Program:**

class Main {

public static void main(String[] args) {

Order[] orders = {

new Order(101, "Arun", 5000),

new Order(102, "kumar", 1400),

new Order(103, "Raja", 7000),

new Order(104,"Niranjan",1100),

new Order(105,"Kishore",1200),

};

Sort sorter=new Sort();

sorter.display(orders);

System.out.println("--Bubble sorting--");

sorter.bubblesort(orders);

sorter.display(orders);

System.out.println("--Quick sorting--");

sorter.quickSort(orders,0,orders.length-1);

sorter.display(orders);

}

public static 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;

}

@Override

public String toString(){

return "OrderId: "+orderid+" Customer Name: "+customerName+" Total Price: "+totalPrice;

}

}

public static class Sort{

public 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+1];

orders[j+1]=orders[j];

orders[j]=temp;

}

}

}

}

public void quickSort(Order[] order, int low, int high) {

if (low < high) {

int pi = partition(order, low, high);

quickSort(order, low, pi - 1);

quickSort(order, pi + 1, high);

}

}

private int partition(Order[] order, int low, int high) {

double pivot = order[high].totalPrice;

int i = low - 1;

for (int j = low; j < high; j++) {

if (order[j].totalPrice <= pivot) {

i++;

Order temp = order[i];

order[i] = order[j];

order[j] = temp;

}

}

Order temp = order[i + 1];

order[i+1] = order[high];

order[high] = temp;

return i + 1;

}

public static void display(Order[] orders) {

for (Order o : orders) {

System.out.println(o);

}

}

}

}

**OUTPUT:**

