**Final Project: Order List Handler for an Online Retailer**

Jordon Paynter

Department of Computer Science, Colorado State University Global

CSC 400: Data Structures and Algorithms

Professor Santosh Kumar Gottipamula

November 5, 2023

Table of Contents

[**Pseudocode** 4](#_Toc150016010)

[**Main Class** 4](#_Toc150016011)

[**Display Class** 7](#_Toc150016012)

[**Class Order** 10](#_Toc150016013)

[**Class OrderInfo** 11](#_Toc150016014)

[**CustomQueue Class** 13](#_Toc150016015)

[Source Code 17](#_Toc150016016)

[Main Class Source code 17](#_Toc150016017)

[Main Class Source code continued. 18](#_Toc150016018)

[Display Class Source code 19](#_Toc150016019)

[Display Class Source code continued. 20](#_Toc150016020)

[Display Class Source code continued 2. 21](#_Toc150016021)

[Order Class Source code continued 23](#_Toc150016022)

[CustomQueue Class Source code 24](#_Toc150016023)

[CustomQueue Class Source code continued. 25](#_Toc150016024)

[**Code Executions** 26](#_Toc150016025)

[Add Order Execution 26](#_Toc150016026)

[Remove Order Execution 27](#_Toc150016027)

[Display Order Execution 28](#_Toc150016028)

[Remove Order by Order number execution for cancellation. 28](#_Toc150016029)

[Exit the Program 29](#_Toc150016030)

# **Pseudocode**

## **Main Class**

Initialize scanner, order, and display

while true:

Display menu options:

1. Add an Order

2. Remove an Order

3. Display Orders

4. Remove order by order number

5. Exit

Get user choice

switch choice:

case 1:

Display "Enter Last Name: "

lastName = validateLastName(scanner)

Display "Enter Order Number: "

orderNumber = validateOrderNumber(scanner)

Display "Enter Total Cost: "

totalCost = validateTotalCost(scanner)

Add order to order queue with lastName, orderNumber, and totalCost

Update and sort arrays in display

Display orders

break

case 2:

Remove order from order queue

Update and sort arrays in display

Display orders

break

case 3:

Display orders from display

break

case 4:

Display "Enter Order Number: "

orderNumberToRemove = validateOrderNumber(scanner)

Remove order from order queue by order number

Update and sort arrays in display

break

case 5:

Display "Exiting the program."

Close scanner

Exit the program

break

default:

Display "Invalid choice. Please enter a valid option."

End Switch

method validateLastName(scanner):

lastNameCheck = scanner.next()

while not lastNameCheck matches "[a-zA-Z]+":

Display "Invalid last name. Please enter a valid last name. Only letters are allowed."

lastNameCheck = scanner.next()

return lastNameCheck

End validateLastName

method validateOrderNumber(scanner):

orderNumberCheck = scanner.nextInt()

while not String.valueOf(orderNumberCheck) matches "[0-9]+":

Display "Invalid order number. Please enter a valid order number using only numbers."

orderNumberCheck = scanner.nextInt()

return orderNumberCheck

End validateOrderNumber

method validateTotalCost(scanner):

totalCostCheck = scanner.nextDouble()

while not String.valueOf(totalCostCheck) matches "[0-9]+.[0-9]{1,2}":

Display "Invalid total cost. Please enter a valid total cost. Only two decimal places are allowed."

totalCostCheck = scanner.nextDouble()

End While

return totalCostCheck

End validateTotalCost

**End Main Class**

## **Display Class**

// Two ArrayLists to store the orders sorted by name and order number

ArrayList<OrderInfo> orderListByName

ArrayList<OrderInfo> orderListByOrderNumber

// Update the ArrayLists and sort them

method updateAndSortArrays(orderQueue):

Clear orderListByName

Clear orderListByOrderNumber

Copy all elements from orderQueue to orderListByName

Copy all elements from orderQueue to orderListByOrderNumber

Sort orderListByName in descending order by last name using quickSort

Sort orderListByOrderNumber in descending order by order number using quickSort

End updateAndSortArrays

// Display the orders sorted by name and order number

method displayOrders():

Print "Orders sorted by name:"

for each order in orderListByName:

Print "Last Name: " + order.getLastName() + ", Order Number: " + order.getOrderNumber() + ", Total Cost: " + order.getTotalCost()

Print "Orders sorted by order number:"

for each order in orderListByOrderNumber:

Print "Last Name: " + order.getLastName() + ", Order Number: " + order.getOrderNumber() + ", Total Cost: " + order.getTotalCost()

End displayOrders

// Quicksort method for sorting the array of OrderInfo objects

method quickSort(orderList, low, high, comparator):

if low < high:

pivotIndex = partition(orderList, low, high, comparator)

quickSort(orderList, low, pivotIndex - 1, comparator)

quickSort(orderList, pivotIndex + 1, high, comparator)

End quickSort

// Partition method for quicksort

method partition(orderList, low, high, comparator):

pivot = orderList[high]

indexSmall = low - 1

for arrayIndex = low to high:

if comparator.compare(orderList[arrayIndex], pivot) <= 0:

indexSmall++

Swap orderList[indexSmall] and orderList[arrayIndex]

Swap orderList[indexSmall + 1] and orderList[high]

return indexSmall + 1

end partition

**End Display Class**

## **Class Order**

Queue<OrderInfo> orderQueue

method addOrder(lastName, orderNumber, totalCost)

Create orderInfo as new OrderInfo(lastName, orderNumber, totalCost)

Add orderInfo to orderQueue

End addOrder

method removeOrder()

if orderQueue is not empty

Remove the front orderInfo from orderQueue

else

Print "Order Queue is empty."

End removeOrder

method removeOrderByOrderNumber(orderNumber)

if orderQueue is not empty

Create tempQueue as empty Queue

orderFound = false

while orderQueue is not empty

orderInfo = Remove the front orderInfo from orderQueue

if orderInfo's orderNumber equals orderNumber

orderFound = true

else

Add orderInfo to tempQueue

if orderFound

Set orderQueue to tempQueue

else

Print "Order not found."

else

Print "Order Queue is empty."

End removeOrderByOrderNumber

method getOrderQueue()

return orderQueue

End getOrderQueue

**End Order Class**

### **Class OrderInfo**

String lastName

int orderNumber

double totalCost

constructor OrderInfo(lastName, orderNumber, totalCost)

Set this.lastName to lastName

Set this.orderNumber to orderNumber

Set this.totalCost to totalCost

End constructor

method getLastName()

return lastName

End getLastName

method getOrderNumber()

return orderNumber

End getOrderNumber

method getTotalCost()

return totalCost

End getTotalCost

**End OrderInfo**

## **CustomQueue Class**

// CustomQueue class to implement a queue using a linked list

class CustomQueue<T>:

private LinkedList<T> elements = new LinkedList<T>

method enqueue(element: T):

elements.addLast(element)

End enqueue

method dequeue(): T:

if not isEmpty():

return elements.removeFirst()

else:

throw IllegalStateException("Queue is empty.")

End dequeue

method isEmpty(): boolean:

return elements.isEmpty()

End isEmpty

method size(): int:

return elements.size()

End size

method peek(): T:

if not isEmpty():

return elements.getFirst()

else:

throw IllegalStateException("Queue is empty.")

End peek

method clear():

elements.clear()

End Clear

method removeOrderByOrderNumber(orderNumber: int):

if not isEmpty():

LinkedList<T> tempQueue = new LinkedList<T>

boolean orderFound = false

while not isEmpty():

T element = dequeue()

if element is OrderInfo:

OrderInfo orderInfo = cast(element)

if orderInfo.getOrderNumber() == orderNumber:

orderFound = true

End if

else:

tempQueue.addLast(element)

End else

if orderFound:

elements = tempQueue

End if

else:

print("Order not found.")

End else

else:

print("Order Queue is empty.")

End else

method getArrayList(): ArrayList<T>:

ArrayList<T> arrayList = new ArrayList<T>

for element in elements:

arrayList.add(element)

End for

return arrayList

End getArrayList

End CustomQueue

# Source Code

**Figure 1a.**

## Main Class Source code

A screenshot of a computer program

Description automatically generated

**Figure 1b.**

## Main Class Source code continued.

A screenshot of a computer program

Description automatically generated

**Figure 2a.**

## Display Class Source code

A screenshot of a computer program

Description automatically generated

**Figure 2b.**

## Display Class Source code continued.

A screenshot of a computer program

Description automatically generated

**Figure 2c.**

## Display Class Source code continued 2.

A computer screen shot of a code

Description automatically generated

**Figure 3a.**

Order Class Source codeA screenshot of a computer program

Description automatically generated

**Figure 3b.**

## Order Class Source code continued

A screen shot of a computer program

Description automatically generated

**Figure 4a.**

## CustomQueue Class Source code

A screenshot of a computer program

Description automatically generated

**Figure 4b.**

## CustomQueue Class Source code continued.

A screenshot of a computer program

Description automatically generated

# **Code Executions**

**Figure 5a.**

## Add Order Execution

A screenshot of a computer program

Description automatically generated

**Figure 5b.**

## Remove Order Execution

A screenshot of a computer

Description automatically generated

**Figure 5b.**

## Display Order Execution

A screenshot of a computer

Description automatically generated

**Figure 5d.**

## Remove Order by Order number execution for cancellation.

A screenshot of a computer program

Description automatically generated

**Figure 5e.**

## Exit the Program

A screenshot of a computer program

Description automatically generated