

# **Software Development II**

Coursework Report 2021/2022

Shanmugaratnam Mohanaranjan

18705841

20200607

### Task 01 – Source Code

```
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.InputMismatchException;
import java.util.Locale;
import java.util.Scanner;
public class CruiseShipMain {
  public static void main(String[] args) {
    CruiseShipCabin[] cabinList = new CruiseShipCabin[12];
    CruiseShipCircularQueue queue = new CruiseShipCircularQueue(12);
    initialiseCabinList(cabinList);
    while (true){
       cruiseMenu(cabinList, queue);
  }
  // cabin list
  public static void initialiseCabinList(CruiseShipCabin[] cabinList) {
    for(int i = 0; i < 12; i++){
       cabinList[i] = new CruiseShipCabin();
       cabinList[i].initializethePassengerList();
    }
  }
  public static void cruiseMenu(CruiseShipCabin[] cabinList, CruiseShipCircularQueue queue){
    Scanner scanner = new Scanner(System.in);
    System.out.println("-----Welcome to Cruise Ship Menu-----");
       // Print out the Menu
    System.out.println(" A : Add customer to Cabin ");
    System.out.println(" V : View all Cabins ");
    System.out.println(" E : Display Empty Cabins ");
    System.out.println(" D : Delete customer from Cabin ");
    System.out.println(" F : Find Cabin from Customer Name ");
    System.out.println(" T : Display passenger expenses ");
    System.out.println(" S : Store program data into a Text file ");
    System.out.println(" L : Load program data from Text file ");
    System.out.println(" O : View passengers ordered alphabetically by name ");
    System.out.println(" Q : Exit ");
    System.out.println("-----");
```

```
String input = scanner.next().toLowerCase(Locale.ROOT);
     switch (input) { // use switch to order from menu
       case "a": addCustomer(cabinList, queue);
         break:
       case "v" : viewCabins(cabinList);
         break:
       case "e" : emptyCabins(cabinList);
         break;
       case "d" : deleteCustomer(cabinList, queue);
         break;
       case "f" : findCustomerCabin(cabinList);
         break;
       case "t" : displayPassengersExpenses(cabinList);
       case "s" : storeData(cabinList);
         break;
       case "1" : loadData();
         break;
       case "o" : alphabeticalPassengerOrder(cabinList);
         break:
       case "q" : exitcabin();
         break;
       default:
         System.out.println(" Invalid input !!!");
  }
  private static boolean isShipFull(CruiseShipCabin[] cabinArray) {
     for(CruiseShipCabin cabin : cabinArray) {
       if(!cabin.isFullBoard()) {
         return false;
       }
    return true;
    // Add Customer to Cabin
  private static void addCustomer(CruiseShipCabin[] cabinArray, CruiseShipCircularQueue
queue) {
    Scanner scanner = new Scanner(System.in);
    while (true){
       try{
          System.out.println("Enter the cabin Number:");
```

```
int cabinNo = scanner.nextInt();
         if (cabinNo == 12)
            break;
         if(isShipFull(cabinArray)) {
            System.out.println("Ship is full. Adding passengers to the waiting list.");
            System.out.println("Enter first name : ");
            String firstName = scanner.next();
            System.out.println("Enter surname: ");
            String surname = scanner.next();
            System.out.println("Enter expenses: ");
            double expenses = scanner.nextDouble();
            queue.newQueue(new CruiseShipPassenger(firstName, surname, expenses));
         } else if (0 <= cabinNo && cabinNo < 12 && !cabinArray[cabinNo].isFullBoard()){
            int passengerNo = 1;
            while (true) {
              boolean endLoop = false;
              System.out.println("Adding passenger" + passengerNo);
              System.out.println("Enter first name: "); // Get first name
              String firstName = scanner.next();
              System.out.println("Enter surname : ");
              String surname = scanner.next();
              System.out.println("Enter expenses : ");
              double expenses = scanner.nextDouble();
              for (int i = 0; i < 3; i++)
                 if(cabinArray[cabinNo].getPassengerList()[i].getFirstName().equals("e")) {
                   cabinArray[cabinNo].getPassengerList()[i].setFirstName(firstName);
                   cabinArray[cabinNo].getPassengerList()[i].setSurname(surname);
                   cabinArray[cabinNo].getPassengerList()[i].setExpenses(expenses);
                   passengerNo++;
                   System.out.println("Do you want to add another passenger into this same
cabin " + cabinNo + "?");
                   System.out.println("Press 'Y' to add another passenger and press any other
key to exit to the previous menu.");
                   scanner.nextLine();
                   String userInput = scanner.nextLine().toLowerCase(Locale.ROOT);
                   if(!userInput.equals("y")) {
                      endLoop = true;
                   break;
              if(endLoop) {
```

```
System.out.println("Leaving cabin " + cabinNo + ".....");
                 break;
          } else if(cabinArray[cabinNo].isFullBoard()) {
            System.out.println("Try another cabin.");
          } else {
            System.out.println("Enter a number between 0 - 12");
       } catch (InputMismatchException x) {
          System.out.println("Enter a number between 0 - 12");
          break;
  }
    // View the Cabins
  public static void viewCabins(CruiseShipCabin[] cabins){
     for(int i = 0; i < 12; i++)
       if(!cabins[i].isEmpty()){
          System.out.println("\nCabin " + i + " : \n");
          for (int j = 0; j < 3; j++)
            CruiseShipPassenger passenger = cabins[i].getPassengerList()[i];
            if(!passenger.getFirstName().equals("e")){
              System.out.println("First Name = " + passenger.getFirstName()); // Print first
name
               System.out.println("Surname = " + passenger.getSurname());
               System.out.println("Expenses = " + passenger.getExpenses() + "\n");
            }
       }else{
          System.out.println("Cabin no :" + i + " is empty");
     }
     // Cabin empty seats
  private static void emptyCabins(CruiseShipCabin[] cabins) {
     for(int i = 0; i < 12; i++) {
       if(!cabins[i].isFullBoard()) {
          for(int j = 0; j < 3; j++) {
            if(cabins[i].getPassengerList()[i].getFirstName().equals("e")) {
               System.out.println("Cabin " + i + " has empty seats.");
              break;
            }
```

```
}
       // Delete Customer from Cabin
  private static void deleteCustomer(CruiseShipCabin[] cabins, CruiseShipCircularQueue
queue) {
     Scanner scanner = new Scanner(System.in);
     System.out.println("Enter the passenger's name");
     String input = scanner.nextLine();
     for(int i = 0; i < 12; i++) {
       for (int j = 0; j < 3; j++) {
if(cabins[i].getPassengerList()[i].getFirstName().toLowerCase(Locale.ROOT).equals(input.toLo
werCase(Locale.ROOT))) {
            CruiseShipPassenger deQueuedPassenger = null;
            if(isShipFull(cabins)) {
              deQueuedPassenger = queue.oldQueue();
            cabins[i].getPassengerList()[j].setFirstName("e");
            cabins[i].getPassengerList()[j].setSurname("e");
            cabins[i].getPassengerList()[i].setExpenses(0);
            System.out.println("Passenger was Deleted");
            if(deQueuedPassenger != null) {
cabins[i].getPassengerList()[j].setFirstName(deQueuedPassenger.getFirstName());
              cabins[i].getPassengerList()[i].setSurname(deQueuedPassenger.getSurname());
              cabins[i].getPassengerList()[j].setExpenses(deQueuedPassenger.getExpenses());
              System.out.println("Passenger added to the ship from the waiting list to the Cruise
Ship");
            break;
       }
  }
    // Find the customer from Cabin
  private static void findCustomerCabin(CruiseShipCabin[] cabins) {
     Scanner scanner = new Scanner(System.in);
     System.out.println("Enter the passenger's name");
     String input = scanner.nextLine();
     boolean isPassengerAvailable = false;
     for(int i = 0; i < 12; i++) {
```

```
for (int i = 0; i < 3; i++) {
if(cabins[i].getPassengerList()[j].getFirstName().toLowerCase(Locale.ROOT).equals(input.toLo
werCase(Locale.ROOT))) {
           System.out.println("Passenger found and the cabin number is "+i);
           isPassengerAvailable = true;
           break:
       }
    if(!isPassengerAvailable) {
       System.out.println("Invalid name passenger");
    // Display Expenses
  private static void displayPassengersExpenses(CruiseShipCabin[] cabins) {
    System.out.println("-----");
    double total = 0;
    for(int i = 0; i < 12; i++) {
       for(int j = 0; j < 3; j++) {
         if(!(cabins[i].getPassengerList()[j].getExpenses() == 0)) {
           System.out.println(cabins[i].getPassengerList()[i].getFirstName() + " - "+
cabins[i].getPassengerList()[j].getExpenses());
           total += cabins[i].getPassengerList()[i].getExpenses();
       }
    System.out.println("-----");
    System.out.println("* Total = "+ total);
    System.out.println("-----"):
  }
    // Store Data in the Text file
  private static void storeData(CruiseShipCabin[] cabins) {
    try {
       FileWriter myWriter = new FileWriter("Data.txt");
       for (int i = 0; i < 12; i++) {
         myWriter.write("Cabin: " + i);
         myWriter.write(System.lineSeparator());
         for (int j = 0; j < cabins[j].passengerList.length; j++) {
           myWriter.write("Passenger " + j + " : ");
           myWriter.write("{ FirstName : " + cabins[i].passengerList[j].getFirstName());
           myWriter.write(", Surname : " + cabins[i].passengerList[j].getSurname());
           myWriter.write(", Expenses: " + cabins[i].passengerList[j].getExpenses() + "}");
```

```
myWriter.write(System.lineSeparator());
       myWriter.close();
       System.out.println("Ship data saved to the text file!");
     } catch (IOException x ) {
       System.out.println("Error occur! + x);
    System.out.println("-----");
 // Load to text file
  private static void loadData() {
    try {
       File file = new File("Data.txt");
       Scanner readFile = new Scanner(file);
       while (readFile.hasNext()){
         System.out.println(readFile.nextLine());
       System.out.println("-----");
     } catch (IOException x) {
       System.out.println("Error occur! + x);
    // Order in the Alphabetical order
  private static void alphabeticalPassengerOrder(CruiseShipCabin[] cabins) {
     int totalCabinPassengers = cabins.length * 3;
    CruiseShipPassenger[] passengerArray = new CruiseShipPassenger[totalCabinPassengers];
     int index = 0:
     for (CruiseShipCabin cabin: cabins) {
       for (int j = 0; j < 3; j++) {
         if (!cabin.getPassengerList()[i].getFirstName().equals("e")) {
            passengerArray[index] = cabin.getPassengerList()[j];
            index++;
       }
    CruiseShipPassenger tempPassenger = null;
     for (int i = 0; i < passengerArray.length; <math>i++) {
       for (int j = i + 1; j < passengerArray.length; <math>j++) {
         if(passengerArray[i]!= null && passengerArray[i]!= null) {
            if (passengerArray[i].getFirstName().compareTo(passengerArray[i].getFirstName())
> 0){
```

```
tempPassenger = passengerArray[i];
    passengerArray[i] = passengerArray[j];
    passengerArray[j] = tempPassenger;
}

}
System.out.println("Display in Alphabetic Order : ");
for (CruiseShipPassenger s : passengerArray) {
    if (s != null && !s.getFirstName().equals("e")) {
        System.out.println(s.getFirstName() + " ");
    }
}

private static void exitcabin() {
    System.exit(0);
}
```

## Task 02 – Source Code

```
public class CruiseShipPassenger {
  String firstName; // Create variables
  String surname;
  double expenses;
  public CruiseShipPassenger(String firstName, String surname, double expenses) {
    this.firstName = firstName;
    this.surname = surname;
    this.expenses = expenses;
  }
  public void setFirstName(String firstName) {
    this.firstName = firstName;
  public void setSurname(String surname) {
    this.surname = surname;
  public void setExpenses(double expenses) {
    this.expenses = expenses;
  public String getFirstName() {
    return firstName;
  }
  public String getSurname() {
    return surname;
  public double getExpenses() {
    return expenses;
```

### Task 03 – Source Code

```
# Check full or not
// Cabin of Crusie Ship
public class CruiseShipCabin {
  CruiseShipPassenger[] passengerList = new CruiseShipPassenger[3]; // Create new list
  public CruiseShipPassenger[] getPassengerList() {
     return passengerList;
  public void initializethePassengerList() { // in list format
     for (int i = 0; i < passengerList.length; <math>i++) {
       passengerList[i] = new CruiseShipPassenger("e", "e", 0);
  }
  public boolean isFullBoard() { // Check of full of board
     for (int i = 0; i < 3; i++) {
       if (!passengerList[i].firstName.equals("e")) {
          return true;
     return false;
  public boolean isEmpty() { // Check of empty board
     for (CruiseShipPassenger passenger: passengerList) {
       if(!passenger.firstName.equals("e")) {
          return false;
    return true;
```

#### # Circular Queue

```
public class CruiseShipCircularQueue {
  private int size, front, rear;
  private CruiseShipPassenger[] queue = new CruiseShipPassenger[12]; // create variable to
queue
  CruiseShipCircularQueue(int size) { // Declare size of queue
     this.size = size;
     this.front = this.rear = -1;
  }
  public CruiseShipPassenger[] getQueue() {
     return queue;
  }
  public void newQueue(CruiseShipPassenger passenger) {
    if ((front == 0 \&\& rear == size - 1) || (rear == (front - 1) % (size - 1))) {
       System.out.print("Waiting list of cruise ship is Full. Sorry! We cannot add anymore
passengers to the waiting list until the cruise ship list is get free .");
     else if (front == -1) {
       front = 0;
       rear = 0;
       queue[rear] = passenger;
     } else if (rear == size - 1 && front != 0) {
       rear = 0;
       queue[rear] = passenger;
     } else {
       rear = (rear + 1);
       if (front <= rear) {
          queue[rear] = passenger;
       }
       else {
          queue[rear] = passenger;
       }
  }
  public CruiseShipPassenger oldQueue() {
```

```
CruiseShipPassenger oldQueuedPassenger;

if (front == -1) {
    return null;
}

oldQueuedPassenger = queue[front];

if (front == rear) {
    front = -1;
    rear = -1;
} else if (front == size - 1) {
    front = 0;
} else {
    front += 1;
}

return oldQueuedPassenger;
}
```

# Task 04 – Testing

Test Case	<b>Expected Result</b>	Actual Result	Pass/Fail
(Cabins Initialised correctly) After program starts, Press 'V'	Displays 'V' for all cabins	Displays 'V' for all cabins	Pass
(Add customer "Mohan" to cabin 5) Select F, enter "Mohan"	Press 'F' Displays "Mohan" for cabin 0	Displays "Mohan" for cabin 0	Pass
Display Total Expenses Press T	Mohan - 234.0 Jana - 345.0 Dilini - 345.0 Jana - 235.0 Kajal - 543.0 Sanu - 987.0 Kajol - 999.0 Raja - 500.0 Rocky - 900.0 Kajalina - 600.0 Kajolina - 723.0 TOM - 555.0 yanu - 232.0 Total = 7198.0	Mohan - 234.0 Jana - 345.0 Dilini - 345.0 Jana - 235.0 Kajal - 543.0 Sanu - 987.0 Kajol - 999.0 Raja - 500.0 Rocky - 900.0 Kajalina - 600.0 Kajolina - 723.0 TOM - 555.0 yanu - 232.0	Pass
Save to Text file Press S	Ship data saved to the text file!	Ship data saved to the text file!	Pass
Exit Press Q	Process finished with exit code 0	Process finished with exit code 0	Pass

## Task 04 - Testing - Discussion

Press V to view all cabins by this to ensure the proper set up to all cabins are created.

Press F then input Mohan by this use of array can be ensure it use like select the name of passenger and search him in which cabin.

Press T to figure out all the expenses by each customer and the net total value can be displayed as final output with each off their value.

Press S to save the inputted value into a text file and array took the storage to it in the text file with more memory.

Press Q to exit from the entire the cabin system.

Using the classes is easy to read and arrays are little bit of confusing in the case of these coding when its good of using arrays is easier while solving the complex problems. Arrays are much help in the case of passenger like sorting and searching them in wide range of 12 cabins with 3 passengers in each of them. While the arrays do the better performance the classes took the low in consuming memories.

# **Self-Evaluation form**

I am completed with my best of knowledge up to task 3.

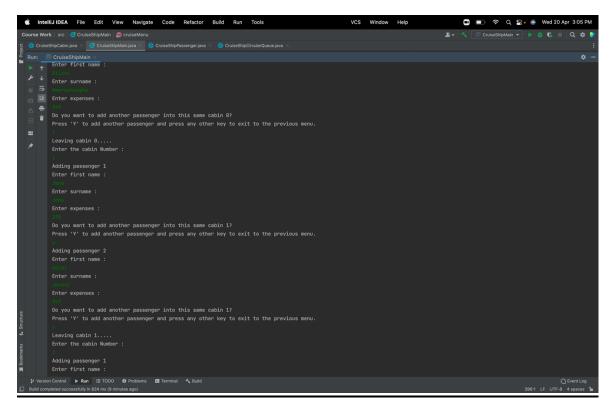
I have face issues in early stage of starting like making the arrays and refer to YouTube and lecture slides recording and google to get my problem to solve. Then again, I got the same problem to make of ques and then find solution from sample codes provided and. Then got from the search of google too.

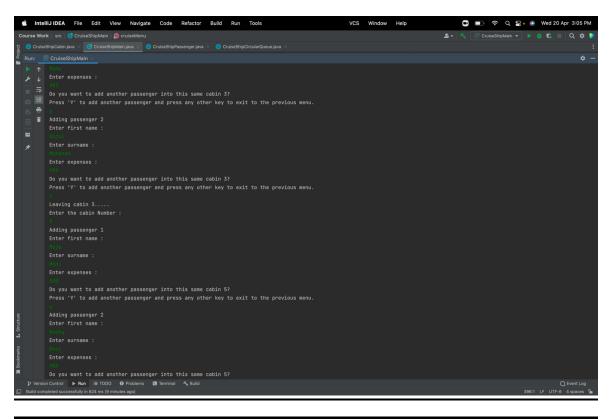
Criteria	Max for Subcomponent	Max Subtotal
<b>Task 1</b> Three marks for each option (A,V,E,D,F,S,L,O) Menu works correctly	24 6	(30)
Task 2 Cabin class correctly implemented. Passenger class correctly implemented. Expenses correctly reported.	14 10 6	(30)
Task 3 Waiting list queue implementation  "A: Add"works correctly  "D: Delete"works correctly  Circular queue implementation	10 3 3 4	(20)
<b>Task 5</b> Test case coverage and reasons Writeup on which version is better and why.	6 4	(10)
Coding Style (Comments, indentation, style) Complete the self-evaluation form indicating what you have accomplished to ensure appropriate feedback.	7 3	(10)
Totals		(100)

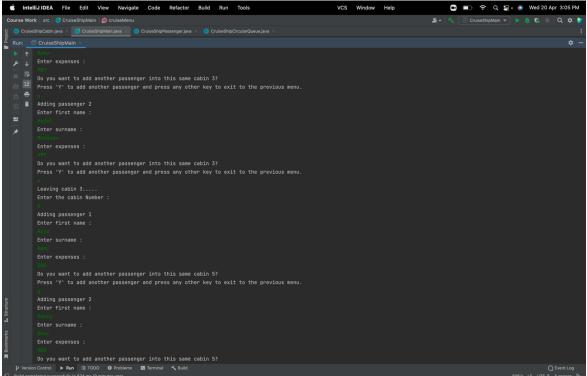
# **Appendices**

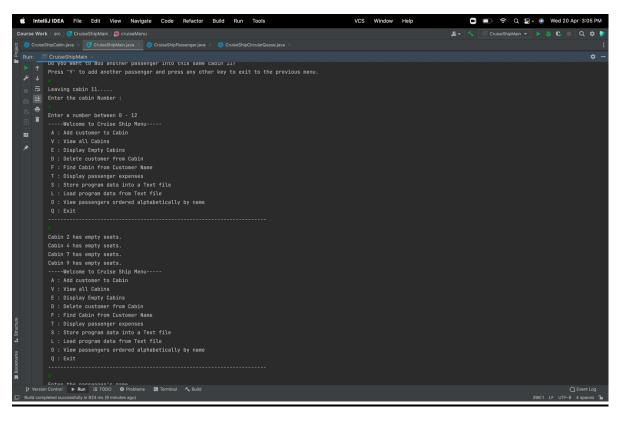
```
** IntellijDEA File Edit View Navigate Code Refactor Build Run Tools VCS Window Help

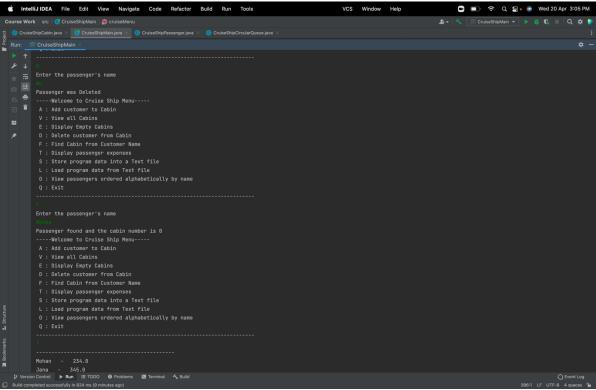
**Procession of Control Manual Procession of C
```



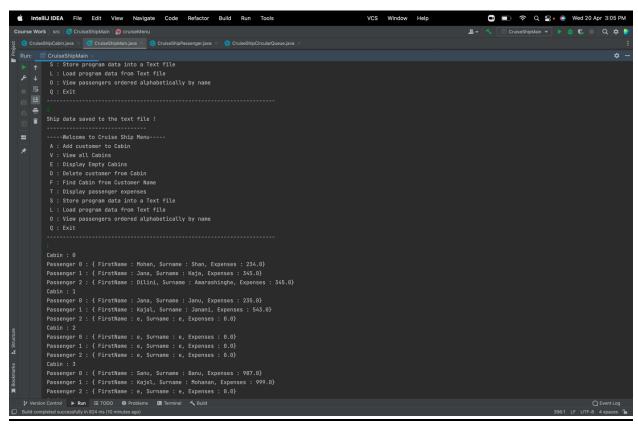




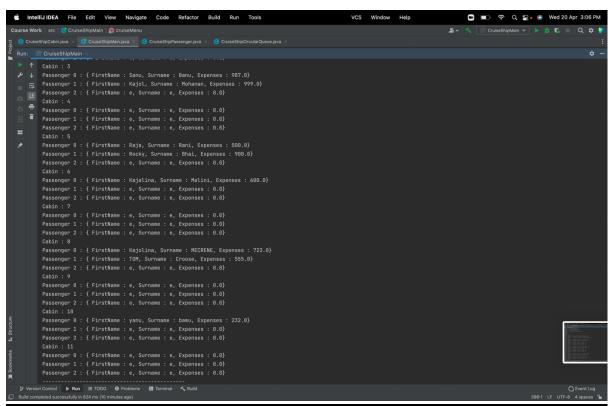


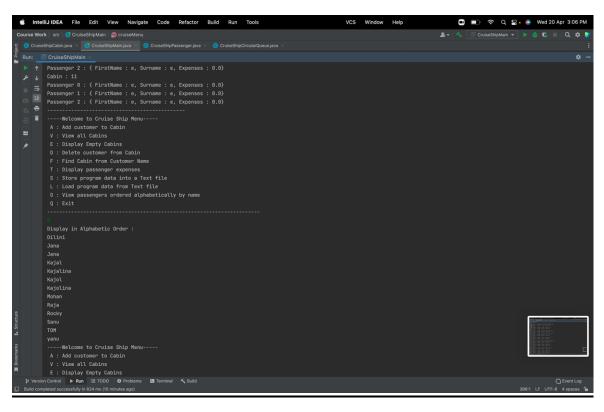


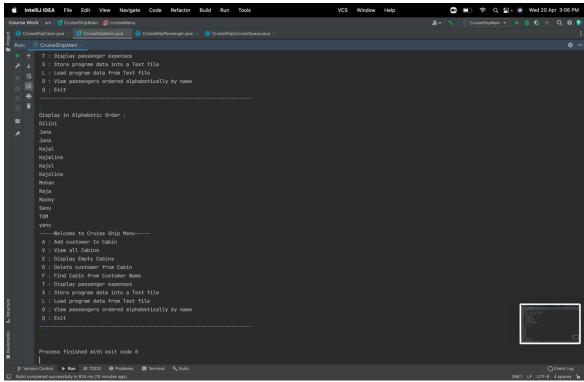
```
intellij IDEA File Edit View Navigate Code Refactor Build Run Tools
                                                                                                                                                                                                                   ■ 🖃 🤝 Q 👺 🕟 Wed 20 Apr 3:05 PM
                                                                                                                                                             VCS Window Help
           ork 
angle src 
angle f G CruiseShipMain 
angle f m g cruiseMe
       ÷
            Mohan - 234.0
Jana - 345.0
Dilini - 345.0
Jana - 235.0
Kajal - 543.0
             Sanu
Kajol
             Kajol - 999.0
Raja - 500.0
             Kajalina - 600.0
Kajolina - 723.0
             TOM - 555.0
yanu - 232.0
              A : Add customer to Cabin
              D : Delete customer from Cabin
              T : Display passenger expenses
S : Store program data into a Text file
             L : Load program data from Text file
O : View passengers ordered alphabetically by name
             Ship data saved to the text file !
            -----Welcome to Cruise Ship Menu-----
n Control > Run III TODO ① Problems 23 Terminal 〈 Build
pleted successfully in 824 ms (9 minutes ago)
```

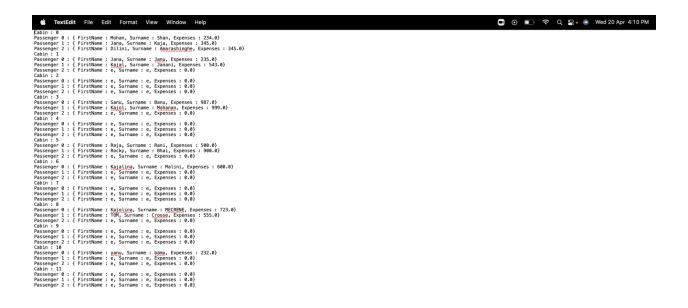


```
intellij IDEA File Edit View Navigate Code Refactor Build Run Tools
                                                                                                                                                                                                                                                                      VCS Window Help
Course Work 
angle src 
angle @ CruiseShipMain 
angle @ cruiseMen
                                                                                                                                                                                                                                                                                                                                      Q # 👂
                          Find Cabin from Customer Name
                 T : Display passenger expenses
                S : Store program data into a Text file
L : Load program data from Text file
         ÷
                Passenger 0 : { FirstName : Mohan, Surname : Shan, Expenses : 234.0}
Passenger 1 : { FirstName : Jana, Surname : Kaja, Expenses : 345.0}
Passenger 2 : { FirstName : Dilini, Surname : Amarashinghe, Expenses : 345.0}
                Passenger 0 : { FirstName : Jana, Surname : Janu, Expenses : 235.0}
Passenger 1 : { FirstName : Kajal, Surname : Janani, Expenses : 543.0}
Passenger 2 : { FirstName : e, Surname : e, Expenses : 0.0}
                Passenger 0 : { FirstName : e, Surname : e, Expenses : 0.0}
Passenger 1 : { FirstName : e, Surname : e, Expenses : 0.0}
Passenger 2 : { FirstName : e, Surname : e, Expenses : 0.0}
                Passenger 1 : { FirstName : Kajol, Surname : Mohanan, Expenses : 999.0}
Passenger 2 : { FirstName : e, Surname : e, Expenses : 8.0}
                Passenger 1 : { FirstName : e, Surname : e, Expenses : 0.0}
Passenger 2 : { FirstName : e, Surname : e, Expenses : 0.0}
                Passenger 1 : { FirstName : Rocky, Surname : Bhai, Expenses : 900.0}
Passenger 2 : { FirstName : e, Surname : e, Expenses : 0.0}
                Passenger 1 : { FirstName : e, Surname : e, Expenses : 0.0}
Passenger 2 : { FirstName : e, Surname : e, Expenses : 0.0}
                n Control ▶ Run Ⅲ TODO ❷ Problems ☑ Terminal 🔨 Build pleted successfully in 824 ms (10 minutes ago)
                                                                                                                                                                                                                                                                                                         ☐ Event Log
396:1 LF UTF-8 4 spaces ¶
```









## **References**

1. Code James (Keep on Coding) ( Accessed on 2022.04.01)

Available at: https://www.youtube.com/watch?v=IUqKuGNasdM&t=31s

2. Alex Lee (Java Arrays Tutorial) (Accessed on 2022.04.03)

Available at: https://www.youtube.com/watch?v=xzjZy-dHHLw

3. Telusco (13 Queue Implementation using Java Part 1 | EnQueue)(Accessed on 2022.04.10)

Available at: <a href="https://www.youtube.com/watch?v=PvDoT79oHTs">https://www.youtube.com/watch?v=PvDoT79oHTs</a>

4. W3schools (2022.04.01)

Available at: <a href="https://www.w3schools.com/java/java">https://www.w3schools.com/java/java</a> arrays.asp

5. Geeksforkers (2022.04.03)

Available at: <a href="https://www.geeksforgeeks.org/array-implementation-of-queue-simple/">https://www.geeksforgeeks.org/array-implementation-of-queue-simple/</a>

6. Kathryn Hodge (Linked in Learning)(18.03.2020)(2022.04.02)

Available at : <a href="https://www.linkedin.com/learning/learning-java-4/welcome-to-learning-java-4/w