

**Software Development II**

Coursework Report 2021/2022

<Warnakula Nadun Kavishka Mendis Wickramanayake >

<w1870591>

<20200928>

**Task 01 – Source Code**

import java.io.FileWriter;

import java.util.\*;

import java.io.File; // Importing the class files

import java.io.IOException;

public class Task1 { // Implementing own sort algorithm

private static String[] cabinSorting(String[] arrayForCabin) {

for (int k = 0; k <arrayForCabin.length ; k++) { //Looping part

for (int y = k + 1; y < arrayForCabin.length; y++) {

if (arrayForCabin[k].compareTo(arrayForCabin[y]) > 0) {

String tempVariable;

tempVariable = arrayForCabin[k];

arrayForCabin[k] = arrayForCabin[y];

arrayForCabin[y] = tempVariable;

}

}

}

for (int k = 0; k < arrayForCabin.length; k++) {

System.out.println(arrayForCabin[k]);

}

return arrayForCabin; //Return the value

}

private static String[] fileLoading(String[] arrayForCabin) {//Creating a methord called fileLoading(to load the files)

File myObject = new File("SavedData.txt");

if (myObject.exists()) {

System.out.println("File name: " + myObject.getName());

System.out.println("Absolute path: " + myObject.getAbsolutePath());

} else {

System.out.println("The file does not exist.");

}

return arrayForCabin; //Return the value

}

private static String[] fileSaveMethord(String[] arrayForCabin) {

//Creating a methord called fileSaveMethord

// (for file saving part)

try {//exception handling

FileWriter dataSaving = new FileWriter("SavedData.txt"); //creating a new object type file writer

for (int k = 0; k < arrayForCabin.length; k++)

{

dataSaving.write("cabin"+k+":"+arrayForCabin[k]+"\n");

}

dataSaving.close();

System.out.println("The file was successfully written to..");

}catch (IOException e) {

System.out.println("There was a mistake.");

e.printStackTrace();

}

return arrayForCabin;//Return the value

}

private static String[] cabinFinder(String[] arrayForCabin) {

System.out.println("Find cabin from customer name");

Scanner input = new Scanner(System.in);

String Name = input.next();

int k;

for (k = 0; k < arrayForCabin.length; k++) {

if (arrayForCabin[k].equals(Name) ) {

System.out.println(k);

}

}

return arrayForCabin;//Return the value

}

private static String[] cabinDeleter(String[] arrayForCabin) {

Scanner input= new Scanner(System.in);

System.out.println("Enter the cabin number you want to remove a passenger");

int k=input.nextInt();

arrayForCabin[k]="Empty";

return arrayForCabin;//Return the value

}

private static String[] cabinsThatEmpty(String[] arrayForCabin) {

System.out.println("Display all empty cabins");

for(int k =0; k<arrayForCabin.length ;k++)

{

if (arrayForCabin[k]=="Empty"){

System.out.println("cabin"+(k)+"-"+arrayForCabin[k]);

}

}

return arrayForCabin;//Return the value

}

private static String[] cabinAdder(String[] arrayForCabin) {

try {

Scanner input = new Scanner(System.in);

System.out.print("Enter a cabin number to add passenger(0-11)");

int k = input.nextInt();

if (arrayForCabin[k] != "Empty") {

System.out.println("A Passenger is in this cabin"); //check is the passenger is in the cabin or not

}

if (arrayForCabin[k] == "Empty") {

System.out.print("Enter Passenger Name:");

arrayForCabin[k] = input.next();

}

return arrayForCabin;//Return the value

}catch (Exception e){

System.out.println("invaide input");

}

return arrayForCabin;//Return the value

}

public static String[] cabinDisplay(String[] arrayForCabin) {

System.out.println("view all cabins");

for (int k = 0; k < arrayForCabin.length ; k++)

{

System.out.println("cabin "+(k)+":"+arrayForCabin[k]);

}

return arrayForCabin;//Return the value

}

public static void main(String[] args) {

String menu;

String[] arrayForCabin = new String[12 ]; //creat array

System.out.println("Enter the following commend:");//given inputs

System.out.println("A - Adds 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("S - Store program data into file");

System.out.println("L - Load program data from file");

System.out.println("O - View passengers Ordered alphabetically by name");

Scanner scanchoice = new Scanner(System.in);

System.out.println();

System.out.println("Enter \"A\", \"V\", \"E\", \"D\", \"F\", \"S\", \"L\" or \"O\" ");

for (int k = 0; k < arrayForCabin.length ; k++)

{

arrayForCabin[k]="Empty"; //initialize cabin elements

}

while(true){

System.out.print("Enter a Input-");

menu = scanchoice.next();

if (menu.equals("V")) {

arrayForCabin=cabinDisplay(arrayForCabin);

} else if (menu.equals("A")) {

arrayForCabin=cabinAdder(arrayForCabin);

} else if (menu.equals("E")) {

arrayForCabin=cabinsThatEmpty(arrayForCabin);

} else if (menu.equals("D")) {

arrayForCabin=cabinDeleter(arrayForCabin);

}else if (menu.equals("F")) {

arrayForCabin=cabinFinder(arrayForCabin);

} else if (menu.equals("S")) {

arrayForCabin=fileSaveMethord(arrayForCabin);

} else if (menu.equals("L")){

arrayForCabin=fileLoading(arrayForCabin);

} else if (menu.equals("O")) {

arrayForCabin=cabinSorting(arrayForCabin);

}

else {

System.out.println("Invalid Input");

}

}

}

}

**Task 02 – Source Code**

**cabin**

import java.util.\*;

public class cabin {

String[] arrayName ={"empty","empty","empty"};

String[] miniArrayName = {"empty","empty","empty"};

int[] moneyArray = new int[3];

static int k=0;

String theName;

String theSurname;

int theExpenses;

String deleteName;

String removeSurname;

int removeExpenses;

public String getName() {

return theName;

}

public String RemoveTheName() {

return deleteName;

}

public int setTheRemoveName() {

Scanner input=new Scanner(System.in);

for(int k=0; k< arrayName.length; k++){

if(arrayName[k]!="empty"){

arrayName[k]="empty";

return 0;

}

}

System.out.println("cabin is full Enter to a new cabin");

return 1;

}

public String getRemoveTheSurname() {

return removeSurname;

}

public void setRemoveTheSurname() {

Scanner input=new Scanner(System.in);

for(int k=0; k< miniArrayName.length; k++){

if(miniArrayName[k]!="empty"){

miniArrayName[k]="empty";

return;

}

}

}

public int getRemoveTheExpenses() {

return removeExpenses;

}

public void setRemoveTheExpenses() {

Scanner input=new Scanner(System.in);

for(int k=0; k< moneyArray.length; k++){

if(moneyArray[k]!=0){

moneyArray[k]=0;

return;

}

}

}

public int settingTheName() {

Scanner input=new Scanner(System.in);

for(int k=0; k< arrayName.length; k++){

if(arrayName[k]=="empty"){

System.out.println("Enter Name");

String theName =input.next();

arrayName[k]=theName;

return 0;

}

}

System.out.println("cabin is full Enter to a new cabin");

return 1;

}

public String getTheSurname() {

return theSurname;

}

public void setTheSurname() {

Scanner input=new Scanner(System.in);

for(int k=0; k< miniArrayName.length; k++){

if(miniArrayName[k]=="empty"){

System.out.println("Enter SurName");

String theSurname=input.next();

miniArrayName[k]=theSurname;

return;

}

}

}

public int getTheExpeneses() {

return theExpenses;

}

public void setTheExpeneses() {

Scanner input=new Scanner(System.in);

for(int k=0; k< moneyArray.length; k++){

if(moneyArray[k]==0){

System.out.println("Enter theExpenses");

int theExpenses=input.nextInt();

moneyArray[k]=theExpenses;

return;

}

}

}

}

# Task2

import java.io.File;

import java.io.FileWriter;

import java.io.IOException;

import java.util.\*;

public class Task2 {

public static int totalAmount;

public static void main(String args[]){

cabin cabin1[]=new cabin[12];

for(int k=0;k< cabin1.length;k++){

cabin1[k]=new cabin();

}

System.out.println("Enter the following commend:");//given inputs

System.out.println("A - Adds 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("S - Store program data into file");

System.out.println("L - Load program data from file");

System.out.println("O - View passengers Ordered alphabetically by name");

System.out.println("T- Print the expenses per passenger");

Scanner choice = new Scanner(System.in);

System.out.println();

System.out.println("Enter \"A\", \"V\", \"E\", \"D\", \"F\", \"S\", \"L\" ,\"O\" or \"T\" ");

while(true){

System.out.print("Enter a Input-");

String displayMenu = choice.next();

if ( displayMenu.equals("V")) {

Passengerviwer(cabin1);

} else if ( displayMenu.equals("A")) {

cabin1 = passengerAdding(cabin1);

} else if ( displayMenu.equals("E")) {

passengerEmptyMethord(cabin1);

} else if ( displayMenu.equals("D")) {

cabin1=passengerDeleting(cabin1);

}else if ( displayMenu.equals("F")) {

PassengerFinding(cabin1);

} else if ( displayMenu.equals("S")) {

cabin1=passengerStore(cabin1);

} else if ( displayMenu.equals("L")){

cabin1=PasengerLoadDetails(cabin1);

} else if ( displayMenu.equals("O")) {

orderOfThePassenger(cabin1);

}

else if ( displayMenu.equals("T")) {

expensesOFThePasenger(cabin1);

}

else {

System.out.println("Invalid Input");

}

}

}

private static void expensesOFThePasenger(cabin[] cabin1) {

for(int k=0;k< cabin1.length;k++){

if(cabin1[k].arrayName[0]!="empty"){

System.out.println("cabin "+k+" passenger "+1+" is "+cabin1[k].moneyArray[0]);

totalAmount+=+cabin1[k].moneyArray[0];

}

if(cabin1[k].arrayName[1]!="empty"){

System.out.println("cabin "+k+" passenger "+2+" is "+cabin1[k].moneyArray[1]);

totalAmount+=+cabin1[k].moneyArray[1];

}

if(cabin1[k].arrayName[2]!="empty"){

System.out.println("cabin "+k+" passenger "+3+" is "+cabin1[k].moneyArray[2]);

totalAmount+=+cabin1[k].moneyArray[2];

}

}

System.out.println("Total is :"+totalAmount);

}

private static void orderOfThePassenger(cabin[] cabin1) {

String[] sortarray=new String[36];

int index=0;

for(int k = 0; k < cabin1.length; k++){

sortarray[index]=cabin1[k].arrayName[0];

index++;

sortarray[index]=cabin1[k].arrayName[1];

index++;

sortarray[index]=cabin1[k].arrayName[2];

index++;

}

for (int k = 0; k <sortarray.length ; k++) {

for (int j = k + 1; j < sortarray.length; j++) {

if (sortarray[k].compareTo(sortarray[j]) > 0) {

String temp;

temp = sortarray[k];

sortarray[k] = sortarray[j];

sortarray[j] = temp;

}

}

}

for (int k = 0; k < sortarray.length; k++) {

if (!sortarray[k].equals("empty")) {

System.out.println(sortarray[k]);

}

}

}

private static cabin[] PasengerLoadDetails(cabin[] cabin1) {

File myObj = new File("SavedData.txt");

if (myObj.exists()) {

System.out.println("File name: " + myObj.getName());

System.out.println("Absolute path: " + myObj.getAbsolutePath());

} else {

System.out.println("The file does not exist.");

}

return cabin1;

}

private static cabin[] passengerStore(cabin[] cabin1) {

try {

FileWriter savedata = new FileWriter("SavedData.txt"); //creat a new object type file writer

for (int k = 0; k < cabin1.length; k++)

{

savedata.write("cabin"+k+"passenger1:"+cabin1[k].arrayName[0]+"\n");

savedata.write("cabin"+k+"passenger2:"+cabin1[k].arrayName[1]+"\n");

savedata.write("cabin"+k+"passenger3:"+cabin1[k].arrayName[2]+"\n");

}

savedata.close();

System.out.println("The file was successfully written to..");

}catch (IOException e) {

System.out.println("There was a mistake.");

e.printStackTrace();

}

return cabin1;

}

private static cabin[] PassengerFinding(cabin[] cabin1) {

Scanner input=new Scanner(System.in);

System.out.print("Enter Passenger Name:");

String name=input.next();

for (int i=0; i<cabin1.length; i++ ){

if(cabin1[i].arrayName[0].equals(name)){

System.out.println("This passenger is in cabin "+i);

} if(cabin1[i].arrayName[1].equals(name)){

System.out.println("This passenger is in cabin "+i);

} if(cabin1[i].arrayName[2].equals(name)){

System.out.println("This passenger is in cabin "+i);

}

}

return cabin1;

}

private static cabin[] passengerDeleting(cabin[] cabin1) {

while (true) {

System.out.println("Enter cabin Number");

Scanner input = new Scanner(System.in);

int cabinnumber = input.nextInt();

int key = cabin1[cabinnumber].setTheRemoveName();

if (key == 0) {

cabin1[cabinnumber].setRemoveTheSurname();

cabin1[cabinnumber].setRemoveTheExpenses();

return cabin1;

}

}

}

private static void passengerEmptyMethord(cabin[] cabin1) {

for(int k=0;k< cabin1.length;k++){

if(cabin1[k].arrayName[0]=="empty"){

System.out.println("cabin "+k+" passenger "+1);

}

if(cabin1[k].arrayName[1]=="empty"){

System.out.println("cabin "+k+" passenger "+2);

}

if(cabin1[k].arrayName[2]=="empty"){

System.out.println("cabin "+k+" passenger "+3);

}

}

}

private static void Passengerviwer(cabin[] cabin1) {

for(int k=0;k< cabin1.length;k++){

if(cabin1[k].arrayName[0]!="empty"){

System.out.println("cabin "+k+" passenger "+1+" is "+cabin1[k].arrayName[0]+" "+cabin1[k].miniArrayName[0]);

}else {

System.out.println("cabin "+k+" passenger "+1+" is empty");

}

if(cabin1[k].arrayName[1]!="empty"){

System.out.println("cabin "+k+" passenger "+2+" is "+cabin1[k].arrayName[1]+" "+cabin1[k].miniArrayName[1]);

}else {

System.out.println("cabin "+k+" passenger "+2+" is empty");

}

if(cabin1[k].arrayName[2]!="empty"){

System.out.println("cabin "+k+" passenger "+3+" is "+cabin1[k].arrayName[2]+" "+cabin1[k].miniArrayName[2]);

}else {

System.out.println("cabin "+k+" passenger "+3+" is empty");

}

}

}

private static cabin[] passengerAdding(cabin[] cabin1) {

while (true){

System.out.println("Enter cabin Number");

Scanner input=new Scanner(System.in);

int cabinnumber=input.nextInt();

int key=cabin1[cabinnumber].settingTheName();

if (key==0){

cabin1[cabinnumber].setTheSurname();

cabin1[cabinnumber].setTheExpeneses();

return cabin1;

}

}

}

}

**Task 03 – Source Code**

**Task 04 – Testing**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| **TEST PLAN for task 1** | | | |
| (Cabins Initialized correctly)  After program starts, Press ‘V’ | Displays ‘empty’ for all cabins | Displays ‘empty’ for all cabins | Pass |
| In case of incorrect letter in addition to the information given in the first menu | Displays” Invalid Input” |  |  |
| (Add customers to cabin 5 “anura”)  (Add customers to cabin 0 to 11)  Select A, enter names | Press ‘v’  Displays names for cabin 0 to 11 | Displays names for cabin 0 to 11 | Pass |
| (Add customer “didula” to cabin 3)  Select A, enter “didula” | Displays “A Passenger is in this cabin” | Displays “A Passenger is in this cabin” | Pass |
| Select E | Displays “No empty cabins” | Displays “No empty cabins” | Pass |
| (Delete customer in cabin 1)  Select D, enter cabin 1 | Press ‘v’  Displays “Empty” for cabin 1 | Displays “Empty” for cabin 1 | Pass |
| (Search customer “anura”)  Select F, enter “anura” | Displays “anura passenger is in cabin 5.” | Displays “anura passenger is in cabin 5.” | Pass |
| Select S | Displays “The file was successfully written to..” | Displays “The file was successfully written to..” | Pass |
| Select L | After typing the letter L, the relevant passengers is displayed in a text file | After typing the letter L, the relevant passengers is displayed in a text file | Pass |
| Select O | The names in alphabetical order | The names in alphabetical order | Pass |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| **TEST PLAN for task 2** | | | |
| (Cabins Initialized correctly)  After program starts, Press ‘V’ | Displays ‘empty’ for all cabins | Displays ‘empty’ for all cabins | Pass |
| In case of incorrect letter in addition to the information given in the first menu | Displays” Invalid Input” |  |  |
| (Add customers to cabin 5 “anura”)  (Add customers to cabin 0 to 11)  Select A, enter names  One or two or three persons in cabins can work as desired in a cabin as required | Press ‘v’  Displays names for cabin 0 to 11  cabin 0 passenger 1 is didula thaveesah  cabin 0 passenger 2 is kamal perera  cabin 0 passenger 3 is empty | Displays names for cabin 0 to 11  cabin 0 passenger 1 is didula thaveesah  cabin 0 passenger 2 is kamal perera  cabin 0 passenger 3 is empty | Pass |
| (Add customer “didula” to cabin 3)  Select A, enter “didula” | Displays “A Passenger is in this cabin” | Displays “A Passenger is in this cabin” | Pass |
| Select E | Displays “No empty cabins” | Displays “No empty cabins” | Pass |
| (Delete customer in cabin 1)  Select D, enter cabin 1 | Press ‘v’  Displays “Empty” for cabin 1 | Displays “Empty” for cabin 1 | Pass |
| (Search customer “anura”)  Select F, enter “anura” | Displays “anura passenger is in cabin 5.” | Displays “anura passenger is in cabin 5.” | Pass |
| Select S | Displays “The file was successfully written to..” | Displays “The file was successfully written to..” | Pass |
| Select L | After typing the letter L, the relevant passengers is displayed in a text file | After typing the letter L, the relevant passengers is displayed in a text file | Pass |
| Select T | Counting the expenses of everyone in the cabin | Counting the expenses of everyone in the cabin | Pass |
| Select O | The names in alphabetical order | The names in alphabetical order | Pass |

**Task 04 – Testing – Discussion**

I tested cases that were related to each of the cases in the first section. Everything on the menu is in perfect working order. All of the test cases, including the sort method, have been finished as shown in the first section. In addition to the first, there are three events in the second segment. Calculate the costs connected with them by putting them in a cabin with two, three, or one other person. Apart from these additions, everything from Part 1 is repeated in Part 2.The final step is to create a waiting list once the cabin is fully booked. Then simply take it from the cabin and put it in the automatic cabin with the first person on the waiting list. studies of cases This is where I show off my menu of add-ons.

To develop part two, I believe that using object class solutions was the best alternative for me. I reasoned that it would be easier to construct and use an object than to call methods created here. I can make an infinite number of things by constraining myself and using object objects. In addition, I got a lot of help from object class solutions for part three.

**Self-Evaluation form**

|  |  |  |
| --- | --- | --- |
| Criteria | Component marks | Expected Mark |
| Task 1 One mark for each option (A,V,E,D,F,S,L,O)  Menu works correctly | 24 6 | 24  6 |
| Student comment: fully implemented and working |  |  |
| Task 2 Cabin class correctly implemented.  Passenger class correctly implemented.  Expenses correctly reported. | 14 10 6 | 14  10  6 |
| Student comment: fully implemented and working |  |  |
| Task 3 Waiting list queue implementation   “A: Add “works correctly   “D: Delete “works correctly  Circular queue implementation | 10 3 3 4 | 3  3  0 |
| Student comment: Not enough time |  |  |
| Task 4 Test case coverage and reasons  Writeup on which version is better and why | 6 4 | 6  4 |
| Student comment: fully implemented and working, I thought the class object was good and I wrote about it |  |  |
| Coding Style (Comments, indentation, style)  Complete the self-evaluation form indicating what you have  accomplished to ensure appropriate feedback. | 7 3 | 7  3 |
| Student comment: fully implemented and working |  |  |
| Totals |  | (100) |
| Demo: At the discretion of your tutor, you may be called on to give a demo of your work to demonstrate understanding of your solutions. If you cannot explain your code and are unable to point to a reference within your code of where this code was found (i.e., in a textbook or on the internet) then significant marks will be lost for that marking component. If you do not attend a requested demo your mark will be capped at 50%. | | |
|  | | |

**References**

**>https://www.w3schools.com/java/java\_files\_create.asp(save data)**