UNIVERSITI TEKNOLOGI MARA

College of Computing, Informatics and Mathematics

CSC305:

PROGRAMMING PARADIGMS

FINAL REPORT:
TICKET CONCERT MANAGEMENT SYSTEM

GROUP:

CDCS1103D

PREPARED BY:

MUHAMMAD AIDIEL BIN MOHAMAD HUSSIN (2022478924)

MUHAMMAD NAZHAN BIN ROZAINI (2022605596)

IRFAN WAFI BIN RAMLI (2022679248)

MUHAMMAD HAZIQ DANISH BIN HAIDIL AMIR (2022866972)

PREPARED FOR:

MADAM SITI 'AISYAH BINTI SA'DAN

DATE SUBMISSION: 22 JANUARY 2024

SESSION: OCTOBER 2023 – FEBRUARY 2024

TABLE OF CONTENT

1.0 GROUP MEMBERS

MUHAMMAD AIDIEL BIN MOHAMAD HUSSIN
(2022478924)
PROJECT MANAGER

IRFAN WAFI BIN RAMLI
(2022679248)
PROGRAMMER 1

MUHAMMAD HAZIQ DANISH BIN HAIDIL AMIR
(2022866972)
PROGRAMMER 2

MUHAMMAD NAZHAN BIN ROZAINI
(2022605596)
PROGRAMMER 3

2.0 DISTRIBUTION OF WORK BETWEEN TEAM MEMBERS

Distribution of works between team members:

	Name	Distribution of works
1.	Muhammad Aidiel bin Mohamad Hussin	• Java Programmer. • Creates Java class definition and main apps for input process. • Manages the documentation during the project.
2.	Muhammad Nazhan bin Rozaini	• Java Programmer. • Creates a method to find a maximum and minimum sales. • Adjust and modify a system interface for both language.
3.	Irfan Wafi bin Ramli	• C Programmer. • Inititate the coding by implementing all the functions such as calcHighest() , calcLowest() , calcPrice() and others. • Debugger if any error occurs in C System.
4.	Muhammad Haziq Danish bin Haidir Amir	• C Programmer. • Establish the remove algorithm, input and output for the system. • System designer for C.

3.0 BASIC PROGRAMMING (ADD, UPDATE AND DELETE)

3.1 C LANGUAGE

1) Add Data

```
printf("\n");
int i;
for( i = 0 ; i < size ; i++ )
{

printf("\nCustomer %d ", i+1);
printf("\nCustomer ID:\n");
scanf("%s", &t[i].custID);
printf("Customer Name:\n");
scanf("%s", &t[i].custName);
printf("Customer Phone Number:\n");
scanf("%s", &t[i].custPhoneNum);
printf("Customer Age:\n");
scanf("%d", &t[i].custAge);
printf("Customer State/City ( Address ):\n");
scanf("%s", &t[i].custAdd);

printf("\nTicket Order %d ", i+1);
printf("\nChoose your preferent Ticket by Entering Concert Code [ 1 - AA001 | 2 - SN001 | 3 - NI001]:\n");
scanf("%s", &t[i].event);

 if( strcmp( t[i].event , "1" ) == 0 )
 {
 strcpy( t[i].event , "Aina Abdul Concert" );
 printf("Enter Event Date you prefer [1|2]:\n");
 scanf("%s", &t[i].date);
     if( strcmp( t[i].date , "1" ) == 0 )
         strcpy(t[i].date,"24/01/2024 (Wednesday) - 8pm");

     else
         strcpy(t[i].date, "28/02/2024 (Wednesday) - 8pm");

 }
 else
 if( strcmp( t[i].event , "2" ) == 0 )
 {
 strcpy( t[i].event , "Siti Nurhaliza Concert" );
 printf("Enter Event Date you prefer [1|2]:\n");
 scanf("%s", &t[i].date);
     if( strcmp( t[i].date , "1" ) == 0 )
         strcpy( t[i].date, "28/01/2024 (Saturday) - 8pm");

     else
         strcpy( t[i].date, "26/01/2024 (Saturday) - 8pm");

 }
 else
 if( strcmp( t[i].event , "3" ) == 0 ){
 strcpy( t[i].event , "Noraniza Idris Concert" );
 printf("Enter Event Date you prefer [1|2]:\n");
 scanf( "%s", &t[i].date);
     if( strcmp( t[i].date , "1" ) == 0 )
         strcpy(t[i].date , "17/01/2024 (Friday) - 8pm");

     else
         strcpy(t[i].date, "14/02/2024 (Friday) - 8pm");

 }

 printf("Enter Seat Section you Prefer [1|2|3]:\n");
 scanf("%s", &t[i].seat );
 if( strcmp( t[i].seat , "1" ) == 0 )
     strcpy( t[i].seat,"CAT 1 (VIP Pass)");
 else
 if( strcmp( t[i].seat , "2" ) == 0 )
     strcpy( t[i].seat,"CAT 2");
 else
 if( strcmp( t[i].seat , "3" ) == 0 )
     strcpy( t[i].seat,"CAT 3");

 printf("\n");
}
```

The user is required to enter the customer details prior to type in ticket details. After that, the ticket details is read as a string data type even though the customer has to input only one character because the same data type will be used to display as an output. For instance, the event attribute will change from “1” which was an input into “Aina Abdul Concert”. We use a string comparison for each condition of if else statement to determine which event, seat or date is selected by the user.

2) Update Data

```
void updateData(struct Ticket t[], int i, int size){
    printf("\n");
    char enterID[100];
    char enterName[100];
    int index = 0;
    printf("Enter customer ID to update phone number:\n");
    scanf("%s", enterID);


    int found = 0;
    for (i = 0; i < size; i++) {
        if ( strcmp(t[i].custID, enterID) == 0 )
        {
            index = i;
            char newPhone[50];
            printf("Customer found!\n\nEnter the new phone number to update:\n");
            scanf("%s", newPhone);
            strcpy(t[i].custPhoneNum, newPhone);
            found = 1;
            break;
        }
    }

    if (found)
    {
        printf("\nPhone number has been updated.\n");
        printf("\nUPDATED DETAILS \n");
        printf("Customer ID: %s\n", t[index].custID);
        printf("Customer Name: %s\n", t[index].custName);
        printf("Phone Number: %s\n", t[index].custPhoneNum);
        printf("Age: %d\n", t[index].custAge);
        printf("Address: %s\n", t[index].custAdd);
        printf("\n");
    } else {
        printf("Customer ID doesn't match or not found. Try again!\n");;
    }
```

To update the contact details of the customer, the function updateData() will be called in the main function where it will start by asking the user to input the Customer ID. Next, it will go through a for loop and go through each data in the array to find the data of the customer that needs to be updated. If it's found, it will then request the user to input the new phone number of the customer and display the updated details. However, if the customer ID does not exist, it will display an appropriate message to the user.

3) Delete Data

```
int removeData(struct Ticket t[], int i , int size){
    char findID[5];
    printf("Enter Customer ID that want to remove:\n");
    scanf("%s", findID);

    int found = 0;
    for(i = 0; i < size; i++){
        if( strcmp(t[i].custID, findID ) == 0){
            found = 1;
            int s;
            for(s = i; s < size-1; s++){
                strcpy(t[s].custID, t[s+1].custID);
                strcpy(t[s].custName, t[s+1].custName);
                t[s].custAge = t[s+1].custAge;
                strcpy(t[s].custAdd, t[s+1].custAdd);
                strcpy(t[s].custPhoneNum, t[s+1].custPhoneNum);
                strcpy( t[s].event , t[s+1].event );
                strcpy(t[s].date, t[s+1].date);
                strcpy( t[s].seat , t[s+1].seat );
            }
            size= size-1;
        }
    }

    if( !found ){
        printf("ID are not in the data. Please enter a valid ID. \n");
    }else {
        printf("Customer found!\nDeleting... \n");
        printf("\nCustomer data for %s is successfully delete.\n", findID);
    }

    return size;
```

To delete a customer's data, the function removeData() is called in the main function. It starts by requesting the user to enter the ID of the customer that needs to be removed. Moving to the process, it will create a loop by going through each data in the array to find the ID of customer. Once found, it will create another for loop where the data of the customer that needs to be removed will be overwritten by the data of the customer next to it and the size of the array will be decrement by one where it will be return to the main function. Finally, it will display a message either the customer data is deleted or not found.

3.2 JAVA LANGUAGE

1) Add Data

```
//Input from console
Scanner input = new Scanner(System.in);
Scanner inputNum = new Scanner(System.in);
Scanner inputText = new Scanner(System.in);
for(int i=0; i<t.length; i++)
{
    System.out.println("\nCustomer " + (i+1) + " ");
    System.out.println("Customer ID: ");
    String ID = inputText.nextLine();
    System.out.println("Customer Name: ");
    String name = inputText.nextLine();
    System.out.println("Customer Phone Number: ");
    String phone = inputText.nextLine();
    System.out.println("Customer Age: ");
    int age = inputNum.nextInt();
    System.out.println("Customer State/City ( Address ): ");
    String address = inputText.nextLine();

    System.out.println("\nTicket Order " + (i+1) + " :");
    System.out.println("Choose your preferent Ticket by Entering Concert Code [1 - AA001|2 - SN001|3 - NI001]: ");
    char ticketID = input.next().charAt(0);
    System.out.println("Enter Event Date you Prefer [1|2]: ");
    char date = input.next().charAt(0);
    System.out.println("Enter Seat Section you Prefer [1|2|3]: ");
    char seat = input.next().charAt(0);

    t[i] = new Ticket(ticketID, date, seat, ID, name, phone, age, address);
}
```

User need to enter their customer ID, customer name, customer phone number, customer age and customer state. After that, the system asks again for users to input their preferent ticket concert by entering ticket concert code. Then, the user needs to choose data concert that they prefer by entering number “1” or “2”. Lastly, users need to enter their preferent seat by entering number “1”, “2” or “3”. All data customer and ticket will be stored into Ticket object in array.

2) Update Data

```
//Search the data of any customer by using their customer ID for reference and update phone number purpose.
System.out.println("\nUPDATE PHONE NUMBER");
Scanner inLine = new Scanner(System.in);
System.out.println("Enter [Y] to continue update phone number or [N] to skip this process: ");
String answer = inLine.nextLine();
while(answer.equalsIgnoreCase("Y"))
{
    System.out.println("Enter customer ID to update phone number: ");
    String searchID = inputText.nextLine();

    int foundIndex = -1;
    //Search for customer by customer ID
    for(int i=0; i<t.length; i++)
    {
        if(t[i].getCustID().equalsIgnoreCase(searchID))
        {
            foundIndex = i;
            break;
        }
    }

    //Update phone number if customer is found
    if (foundIndex != -1)
    {
        System.out.println("Customer found!\n\nEnter the new phone number to update: ");
        String newPhoneNum = inputText.nextLine();

        //Update the new phone number for customer
        t[foundIndex].setCustPhoneNum(newPhoneNum);

        System.out.println("\nPhone number has been updated.");
        System.out.println("\nUpdated details:" + t[foundIndex].toStringCustomer());
    }
    else
    {
        System.out.println("Customer ID doesn't match or not found. Try again!");
    }
    System.out.println("\nEnter [Y] to continue update phone number or [N] to skip this process: ");
    answer = inLine.nextLine();
}
```

User will be asked if they want to update the number phone or not. If user input “Y”, system will ask user to enter customer ID that want to update the phone number. If user enters correct customer ID, system will print message update was successfully and print new details of data based on customer ID that user changed and system ask again if user want to update another customer phone number or not. System prompt error message if user enter invalid customer ID. If user enter “N”, system will skip this update process.

3) Delete Data

```
//Remove data
System.out.println("\nREMOVE CUSTOMER ORDER");
System.out.println("Enter [Y] to Continue OR [N] to skip this process: ");
String remove = inLine.nextLine();
while(remove.equalsIgnoreCase("Y"))
{
    System.out.println("Enter Customer ID that Want to Remove: ");
    String searchIDRemove = inputText.nextLine();

    int foundIndex = -1;
    //Search for customer by customerID
    for(int i=0; i<t.length; i++)
    {
        if(t[i].getCustID().contains(searchIDRemove))
        {
            foundIndex = i;
            break;
        }
    }

    //Remove customer if customerID is found
    if (foundIndex != -1)
    {
        System.out.println("Customer found!\nDeleting... \n");

        //Remove customer
        t[foundIndex] = new Ticket();
        t[foundIndex] = null;

        System.out.println("Customer data for "+searchIDRemove+ " is successfully delete.");
    }
    else
    {
        System.out.println("Customer ID not found. Try again");
    }
    System.out.println("\nEnter [Y] to Continue OR [N] to skip this process: ");
    remove = inLine.nextLine();
}
```

For removing data, the process is same as update data. Basically, user will be asked if they want to remove a customer data or not. If user input “Y”, the system will ask user to enter customer ID that user want to delete. If user enter correct customer ID, system will proceed to delete the customer data and then system ask again if user want to delete another customer data or not. The prompt error will appear if user enter the wrong customer ID and system will ask again if want to delete customer data or not. If user enter “N”, system will skip this removing process.

4.0 PROBLEM TO BE SOLVED

Artists and musicians across the globe have been making performances for their beloved fans to perform their album or their latest singles. They would usually perform at each continent of the world and fans would go out of their way to do any means to meet their idol. To them it's the only form of connection and meaningful bond they will attain to which it's not a surprise that most of the events are sold out. For beginners, purchasing a ticket to a concert is much harder than they expected. When planning the concert, most of the time the assigned venue is held for a smaller audience than the fanbase originally is which demonstrates how intense the competition is for purchasing these tickets.

There are various issues that individuals would complain or voice out when it comes to buying a ticket. A common one would be that they have to go out to the closest ticket stations respectively. Fans with no other option would have to stand in a very long queue which could end up for more than two hours if not more which is not appropriate since it would make the experience stressful, melancholic and time wasting. To add it up, fans would even accidentally loss or damage their own tickets. This may happen because of the printing material is cheap as well as not waterproof which could get soaked and torn up easily. Physical tickets may lead to be stolen or mistakenly thrown away. Even worst, there have been occasion where customers would face strangers persuading to buy the tickets off them for twice the amount or even higher. With that, customers should always be given privacy and security to their success in getting tickets.

As a solution, our team used our expertise to create a ticket managing system to handle all customer's data accurately and hand out tickets accordingly. The system will be simple to navigate and easy to use for all ages implying that it is user-friendly. In our eyes, this system provides a structure and organization which allows the administrator to handle each ticket or events effectively and allows each data to be handle secured. The program will have the following processes:

i. Calculate the **sum** of all earnings from each event based on ticket ID.
ii. **Count** the total attendees for each section.
iii. Determine the **maximum** sales and attendees gained by comparing each event thoroughly.
iv. Determine the **minimum** sales and attendees gained by comparing each event thoroughly.
v. **Search** the data of any customer by using their ID and name for phone number update purpose.
vi. **Remove** the data of any customer by using their customer ID.

4.1 C Language

i. Calculate the **sum** of all earnings from each event based on ticket ID.

```
//Calculate the sum of all earnings from each event based on their name
void calcTotalSales( struct Ticket t[] , int size )
{
    int i;
    float sumAA = 0.00;
    float sumSN = 0.00;
    float sumNI = 0.00;

    for ( i = 0; i < size ; i++ )
    {
    if( strcmp( t[i].event , "Aina Abdul Concert" ) == 0 )
            sumAA = sumAA + calcPrice( t , i );
    else
    if( strcmp( t[i].event , "Siti Nurhaliza Concert" ) == 0  )
        sumSN = sumSN + calcPrice( t , i  );
    else
    if( strcmp( t[i].event , "Noraniza Idris Concert" ) == 0 )
        sumNI = sumNI + calcPrice( t , i  );
    }

    printf("\nTotal Earnings for Each Concert: ");
    printf("\n\tAina Abdul's Concert : RM%.2f", sumAA);
    printf("\n\tSiti Nurhaliza's Concert : RM%.2f", sumSN);
    printf("\n\tNoraniza Idris's Concert : RM%.2f", sumNI);
}
```

To calculate the summation of all earnings from each event based on ticket ID, we use a function named *calcTotalSales* that send all the attributes from struct named Ticket and integer size which is the size of the array. Inside the function definition, it creates three variables to store all the total sales of each event. For example, *sumAA* stores the total sales for Aina Abdul's concert, *sumSN* stores the total sales for Siti Nurhaliza concert and *sumNI* stores the total sales for Noraniza Idris's concert. The function contains a for loop to get all the array of the attribute from struct Ticket and count all the total sales based on each event using another function called *calcPrice*.

ii. **Count** the total attendees for each section.

```
//Count the total attendees for each section
void countSection( struct Ticket t[] , int size )
{
    int i;
    int catOne = 0;
    int catTwo = 0;
    int catThree = 0;

    for (i = 0; i < size; i++)
    {
        if( strcmp( t[i].seat , "CAT 1 (VIP Pass)" ) == 0 )
            catOne = catOne + 1;
        else
        if( strcmp( t[i].seat , "CAT 2" ) == 0 )
            catTwo = catTwo + 1;
        else
        if( strcmp( t[i].seat , "CAT 3" ) == 0 )
            catThree = catThree + 1;
    }

    printf("\nTotal customer that chose Cat One : %d", catOne);
    printf("\nTotal customer that chose Cat Two : %d", catTwo);
    printf("\nTotal customer that chose Cat Three : %d", catThree);

}
```

The system use a function called *countSection* to calculate all the total attendees on each section. Similarly with function *calcTotalSales*, the for loop receive each struct Ticket array to get all the attributes and get the attribute seat to calculate and stores the total attendees. After finishing the calculation, it will display all the total attendees in each seat section.

iii. Determine the **maximum** sales and attendees gained by comparing each event thoroughly.

```
//Determine the maximum sales and attendees gained by comparing each event thoroughly
void calcHighest( struct Ticket t[] , int size )
{
    int i;
    int countAA = 0;
    int countSN = 0;
    int countNI = 0;
    int highestAtt = 0;
    float highestSales = 0.00;
    float sumAA = 0.00;
    float sumSN = 0.00;
    float sumNI = 0.00;
    char SalesName[50];
    char AttName[50];

    for ( i = 0; i < size; i++ )
    {
        if( strcmp( t[i].event , "Aina Abdul Concert" ) == 0  )
            {
                sumAA = sumAA + calcPrice( t , i );
                countAA = countAA + 1;
            }
        else
        if( strcmp( t[i].event , "Siti Nurhaliza Concert" ) == 0 )
            {
                sumSN = sumSN + calcPrice( t , i );
                countSN = countSN + 1;
            }
        else
        if( strcmp( t[i].event , "Noraniza Idris Concert" ) == 0 )
            {
                sumNI = sumNI + calcPrice( t , i );
                countNI = countNI + 1;
            }
    }

    if ( sumAA > highestSales )
        {
          highestSales = sumAA;
          strcpy( SalesName , "Aina Abdul's Concert");
        }
    if ( sumSN > highestSales )
        {
          highestSales = sumSN;
          strcpy( SalesName,"Siti Nurhaliza's' Concert");
        }
    if ( sumNI > highestSales )
        {
          highestSales = sumNI;
          strcpy( SalesName , "Noraniza Idris's' Concert");
        }

    if ( countAA > highestAtt )
        {
         highestAtt = countAA;
         strcpy( AttName , "Aina Abdul's Concert" );
        }
    if ( countSN > highestAtt )
        {
          highestAtt = countSN;
          strcpy( AttName , "Siti Nurhaliza's Concert" );
        }
    if ( countNI > highestAtt )
        {
          highestAtt = countNI;
          strcpy( AttName , "Noraniza Idris's Concert" );
        }

    printf("\nName of highest attended event : %s", AttName );
    printf("\nTotal attendees for the highest event : %d", highestAtt);

    printf("\nName of highest sales gained from an event : %s", SalesName);
    printf("\nTotal sales for the highest event : %.2f", highestSales);

}
```

To determine the maximum sales and attendees, the function called *calcHighest* contain a for loop to get each array of struct Ticket and calculate all the sales and the total attendees of each event, the for loop also determined the highest sales by comparing if the total sales of the event is higher than current value of the variable named *highestSales*. Right after that, the for loop also determined the highest attended event by comparing if the total attendees is higher than the value hold by the variable named *highestAtt*. When the for loop is done, it will display the highest attended event with the total of the attendees and the highest sales gained from an event with its total sales.

iv. Determine the **minimum** sales and attendees gained by comparing each event thoroughly.

```
//Determine the lowest sales and attendees gained by comparing each event thoroughly
void calcLowest( struct Ticket t[] , int size )
{
    int i;
    int countAA = 0;
    int countSN = 0;
    int countNI = 0;
    int lowestAtt = 999;
    float lowestSales = 999999.00;
    float sumAA = 0.00;
    float sumSN = 0.00;
    float sumNI = 0.00;
    char SalesName[50];
    char AttName[50];

    for (i = 0; i < size; i++ )
    {
        if( strcmp( t[i].event , "Aina Abdul Concert" ) == 0 )
            {
                sumAA = sumAA + calcPrice( t , i );
                countAA = countAA + 1;
            }
        else
        if( strcmp( t[i].event , "Siti Nurhaliza Concert" ) == 0 )
            {
                sumSN = sumSN + calcPrice( t , i );
                countSN = countSN + 1;
            }
        else
        if( strcmp( t[i].event , "Noraniza Idris Concert" ) == 0 )
            {
                sumNI = sumNI + calcPrice( t , i );
                countNI = countNI + 1;
            }
    }

    if ( sumAA < lowestSales )
        {
          lowestSales = sumAA;
          strcpy( SalesName , "Aina Abdul's Concert");
        }
    if ( sumSN < lowestSales )
        {
          lowestSales = sumSN;
          strcpy( SalesName , "Siti Nurhaliza's Concert");
        }
    if ( sumNI < lowestSales )
        {
          lowestSales = sumNI;
          strcpy( SalesName , "Noraniza Idris's Concert");
        }

    if ( countAA < lowestAtt )
        {
         lowestAtt = countAA;
         strcpy( AttName , "Aina Abdul's Concert" );
        }
    if ( countSN < lowestAtt )
        {
          lowestAtt = countSN;
          strcpy( AttName , "Siti Nurhaliza's Concert" );
        }
    if ( countNI < lowestAtt )
        {
          lowestAtt = countNI;
          strcpy( AttName , "Noraniza Idris's Concert");
        }

    printf("\nName of lowest attended event : %s", AttName);
    printf("\nTotal attendees for the lowest event : %d", lowestAtt);

    printf("\nName of lowest sales gained from an event : %s", SalesName);
    printf("\nTotal sales for the lowest event : %.2f", lowestSales);

}
```

The system determines the minimum sales and attendees by using a function called *calcLowest*. In this function, it works similarly with the function *calcHighest* with calculating the total sales and total attendees of each event first. But when it comes to comparing, it will compare if the total sales of the event is lower than the variable called *lowestSales* and compares the total attendees if the total attendees is lower than the variable called *lowestAtt*. Outside the loop, it will display the information regarding the minimum sales and attendees.

v. **Search** the data of any customer by using their ID and name for phone number update purpose.

```
void updateData(struct Ticket t[], int i, int size){
    printf("\n");
    char enterID[100];
    char enterName[100];
    int index = 0;
    printf("Enter customer ID to update phone number:\n");
    scanf("%s", enterID);


    int found = 0;
    for (i = 0; i < size; i++) {
        if ( strcmp(t[i].custID, enterID) == 0 )
        {
            index = i;
            char newPhone[50];
            printf("Customer found!\n\nEnter the new phone number to update:\n");
            scanf("%s", newPhone);
            strcpy(t[i].custPhoneNum, newPhone);
            found = 1;
            break;
        }
    }

    if (found)
    {
        printf("\nPhone number has been updated.\n");
        printf("\nUpdated details: \n");
        printf("Customer ID: %s\n", t[index].custID);
        printf("Customer Name: %s\n", t[index].custName);
        printf("Phone Number: %s\n", t[index].custPhoneNum);
        printf("Age: %d\n", t[index].custAge);
        printf("Address: %s\n", t[index].custAdd);
        printf("\n");
    } else {
        printf("Customer ID doesn't match or not found. Try again!\n");;
    }

}
```

When it comes to searching for data to update, it used the function called *updatedData*. The function will prompt the user to enter the desired customer ID to update the phone number. The for loop will run through each struct Ticket array to find if the customer ID entered is exist in the array. If it is found, it will prompt the user to enter the new phone number and display all the attribute of the customer ID. Otherwise it will display a message where it says the ID doesn’t exist inside the array.

vi. **Remove** the data of any customer by using their customer ID.

```
int removeData(struct Ticket t[], int i , int size){
    char findID[5];
    printf("Enter Customer ID that want to remove:\n");
    scanf("%s", findID);

    int found = 0;
    for(i = 0; i < size; i++){
        if( strcmp(t[i].custID, findID ) == 0){
            found = 1;
            int s;
            for(s = i; s < size-1; s++){
                strcpy(t[s].custID, t[s+1].custID);
                strcpy(t[s].custName, t[s+1].custName);
                t[s].custAge = t[s+1].custAge;
                strcpy(t[s].custAdd, t[s+1].custAdd);
                strcpy(t[s].custPhoneNum, t[s+1].custPhoneNum);
                strcpy( t[s].event , t[s+1].event );
                strcpy(t[s].date, t[s+1].date);
                strcpy( t[s].seat , t[s+1].seat );
            }
            size= size-1;
        }
    }

    if( !found ){
         printf("ID are not in the data. Please enter a valid ID. \n");
    }else {
        printf("Customer found!\nDeleting... \n");
        printf("\nCustomer data for %s is successfully delete.\n", findID);
        return size;
    }

}
```

The remove method exist inside a function called *removeData.* It will prompt the user to enter the customer ID that want to be remove. If the customer ID exist inside the array of struct Ticket, the current index of the array will copy the next array and will continue this cycle until the for loop is ended. After the copying process is done, it will deduct the size of the array by one to be display later and the function will return the value of variable size. If the customer ID does not exist, it will appear a message where the ID is not in the data.

4.2 Java Language

i. Calculate the **sum** of all earnings from each event based on ticket ID.

```
//Calculate the sum of all earnings from each event based on ticket ID.
double sumAA = 0, sumSN = 0, sumNI = 0;
for(int i=0; i<t.length; i++)
{
    if(t[i].getTicketID() == '1')
        sumAA += t[i].calcPrice();
    else if(t[i].getTicketID() == '2')
        sumSN += t[i].calcPrice();
    else if(t[i].getTicketID() == '3')
        sumNI += t[i].calcPrice();
}
System.out.println("\nTotal Earnings for Each Concert: ");
System.out.println("\tAina Abdul's Concert: RM " + sumAA);
System.out.println("\tSiti Nurhaliza's Concert: RM " + sumSN);
System.out.println("\tNoraniza Idris's Concert: RM " + sumNI);
```

To find sum of all earnings for all event, in for loop, system will call ticket ID from Ticket object in array by using getter caller and do the comparison if ticket ID is same with conditional statement or not. For example, if conditional statement is true for ticket ID equals to 1, system will do an update of *sumAA* value by calling *calcPrice()* method. This process will do same process if ticket ID is equal to 2 or 3. Then, the process will be repeating until the for loop conditional statement is false. After that, system will print total earnings for each concert.

ii. **Count** the total attendees for each section.

```
//Count the total attendees for each section.
int countAA = 0, countSN = 0, countNI = 0;
for(int i=0; i<t.length; i++)
{
    if(t[i].getTicketID() == '1')
        countAA++;
    else if(t[i].getTicketID() == '2')
        countSN++;
    else if(t[i].getTicketID() == '3')
        countNI++;
}
System.out.println("\nTotal Attendees for Each Concert: ");
System.out.println("\tAina Abdul's Concert: " + countAA);
System.out.println("\tSiti Nurhaliza's Concert: " + countSN);
System.out.println("\tNoraniza Idris's Concert: " + countNI);
```

To find total of all attendees for all event, in for loop, system will call ticket ID from Ticket object in array by using getter caller and do the comparison if ticket ID is same with conditional statement or not. For example, if conditional statement is true for ticket ID equals to 1, system will do an update of *countAA* value by increasing value for *countAA* into 1. This process will do same process if ticket ID is equal to 2 or 3. Then, the process will be repeating until the for loop conditional statement is false. After that, system will print total attendees for each concert.

iii. Determine the **maximum** sales and attendees gained by comparing each event thoroughly.

```
//Determine the maximum sales and attendees gained by comparing each event thoroughly .
int countMax = 0;
double maxSales = 0;
String maxSalesEvent = "", maxCountEvent = "";
for(int i=0; i<t.length; i++)
{
    if(sumAA > maxSales)
    {
        maxSales = sumAA;
        maxSalesEvent = "Aina Abdul's Concert";
    }

    if(sumSN > maxSales)
    {
        maxSales = sumSN;
        maxSalesEvent = "Siti Nurhaliza's Concert";
    }

    if(sumNI > maxSales)
    {
        maxSales = sumNI;
        maxSalesEvent = "Noraniza Idris's Concert";
    }


    if(countAA > countMax)
    {
        countMax = countAA;
        maxCountEvent = "Aina Abdul's Concert";
    }

    if(countSN > countMax)
    {
        countMax = countSN;
        maxCountEvent = "Siti Nurhaliza's Concert";
    }

    if(countNI > countMax)
    {
        countMax = countNI;
        maxCountEvent = "Noraniza Idris's Concert";
    }
}
System.out.println("\nName of highest attended event: " +maxCountEvent);
System.out.println("Total attendees for the highest event: " +countMax);
System.out.println("Name of highest sales gained from an event: " +maxSalesEvent);
System.out.println("Total sales for highest sales event: RM " +maxSales);
```

To find maximum sales based on three events, in for loop, the system will do comparison for each total earnings for all concert first and followed by do comparison for each total of attendees. If the conditional statement for comparison of total earnings is true, the system will set *maxSales* into total earnings for that event and assign *maxEventSales* to current highest concert event name. Next, system will do comparison for total attendees. If the conditional statement for comparison of total attendees is true, the system will set *countMax* into attendees for current event and assign *maxCountEvent* into current highest concert event. Then, the process will repeat until the conditional statement for loop is false. After that, the system will print event name for maximum attendees, total attendees for maximum attendees, event name for maximum sales and total price for maximum sales.

iv. Determine the **minimum** sales and attendees gained by comparing each event thoroughly.

```
//Determine the minimum sales and attendees gained by comparing each event thoroughly
int countMin = 99999;
double minSales = 999999.99;
String minSalesEvent = "", minCountEvent = "";
for(int i=0; i<t.length; i++)
{
    if(sumAA < minSales)
    {
        minSales = sumAA;
        minSalesEvent = "Aina Abdul's Concert";
    }

    if(sumSN < minSales)
    {
        minSales = sumSN;
        minSalesEvent = "Siti Nurhaliza's Concert";
    }

    if(sumNI < minSales)
    {
        minSales = sumNI;
        minSalesEvent = "Noraniza Idris's Concert";
    }

    if(countAA < countMin)
    {
        countMax = countAA;
        minCountEvent = "Aina Abdul's Concert";
    }

    if(countSN < countMax)
    {
        countMin = countSN;
        minCountEvent = "Siti Nurhaliza's Concert";
    }

    if(countNI < countMin)
    {
        countMin = countNI;
        minCountEvent = "Noraniza Idris's Concert";
    }
}
System.out.println("\nName of lowest attended event: " +minCountEvent);
System.out.println("Total attendees for the lowest event: " +countMin);
System.out.println("Name of lowest sales gained from an event: " +minSalesEvent);
System.out.println("Total sales for lowest sales event: RM " +minSales);
```

To find minimum sales based on three events, in for loop, the system will do comparison for each total earnings for all concert first and followed by do comparison for each total of attendees. If the conditional statement for comparison of total earnings is true, the system will set *minSales* into total earnings for that event and assign *minEventSales* to current lowest concert event name. Next, system will do comparison for total attendees. If the conditional statement for comparison of total attendees is true, the system will set *countMin* into attendees for current event and assign *minCountEvent* into current lowest concert event. Then, the process will repeat until the conditional statement for loop is false. After that, the system will print event name for minimum attendees, total attendees for minimum attendees, event name for minimum sales and total price for minimum sales.

v. **Search** the data of any customer by using their ID and name for phone number update purpose.

```
//Search the data of any customer by using their customer ID for reference and update phone number purpose.
System.out.println("\nUPDATE PHONE NUMBER");
Scanner inLine = new Scanner(System.in);
System.out.println("Enter [Y] to continue update phone number or [N] to skip this process: ");
String answer = inLine.nextLine();
while(answer.equalsIgnoreCase("Y"))
{
    System.out.println("Enter customer ID to update phone number: ");
    String searchID = inputText.nextLine();

    int foundIndex = -1;
    //Search for customer by customer ID
    for(int i=0; i<t.length; i++)
    {
        if(t[i].getCustID().equalsIgnoreCase(searchID))
        {
            foundIndex = i;
            break;
        }
    }

    //Update phone number if customer is found
    if (foundIndex != -1)
    {
        System.out.println("Customer found!\n\nEnter the new phone number to update: ");
        String newPhoneNum = inputText.nextLine();

        //Update the new phone number for customer
        t[foundIndex].setCustPhoneNum(newPhoneNum);

        System.out.println("\nPhone number has been updated.");
        System.out.println("\nUpdated details:" + t[foundIndex].toStringCustomer());
    }
    else
    {
        System.out.println("Customer ID doesn't match or not found. Try again!");
    }
    System.out.println("\nEnter [Y] to continue update phone number or [N] to skip this process: ");
    answer = inLine.nextLine();
}
```

To update the customer's phone number, in a while loop, user will be asked if they want to update the number phone or not. If user input "Y", system will ask user to enter customer ID that want to update the phone number. If user enters correct customer ID, system will print message update was successfully and print new details of data based on customer ID that user changed, and system ask again if user want to update another customer phone number or not. System prompt error message if user enter invalid customer ID. If user enter "N", system will skip this update process.

vi. **Remove** the data of any customer by using their customer ID.

```
//Remove data
System.out.println("\nREMOVE CUSTOMER ORDER");
System.out.println("Enter [Y] to Continue OR [N] to skip this process: ");
String remove = inLine.nextLine();
while(remove.equalsIgnoreCase("Y"))
{
    System.out.println("Enter Customer ID that Want to Remove: ");
    String searchIDRemove = inputText.nextLine();

    int foundIndex = -1;
    //Search for customer by customerID
    for(int i=0; i<t.length; i++)
    {
        if(t[i].getCustID().contains(searchIDRemove))
        {
            foundIndex = i;
            break;
        }
    }

    //Remove customer if customerID is found
    if (foundIndex != -1)
    {
        System.out.println("Customer found!\nDeleting... \n");

        //Remove customer
        t[foundIndex] = new Ticket();
        t[foundIndex] = null;

        System.out.println("Customer data for "+searchIDRemove+ " is successfully delete.");
    }
    else
    {
        System.out.println("Customer ID not found. Try again");
    }
    System.out.println("\nEnter [Y] to Continue OR [N] to skip this process: ");
    remove = inLine.nextLine();
}
```

To remove the customer's data, in a while loop, the process is same as update data. Basically, user will be asked if they want to remove a customer data or not. If user input "Y", the system will ask user to enter customer ID that user want to delete. If user enter correct customer ID, system will proceed to delete the customer data and then system ask again if user want to delete another customer data or not. The prompt error will appear if user enter the wrong customer ID and system will ask again if want to delete customer data or not. If user enter "N", system will skip this removing proces

5.0 SUGGESTED SOLUTION

There are two programming language that will be used for this project:

5.1 C Language

The C language is a compiled language that converts the code into machine language so that it can be understood by the machine or system. C is a low-level language. It has difficult interpretation for the user, but it has a closer significance to the machine level code. Call by value and call by reference are supported in C. C language has a simple syntax that is relatively easy to use and apply in this project.

5.1.1 Struct Definition

```
struct Ticket {
    char custID[5];
    char custName[50];
    char custPhoneNum[15];
    int custAge;
    char custAdd[100];
    char event[50];
    char date[50];
    char seat[50];
}
```

5.1.2 List of Array Used

- `struct Ticket t[size];`

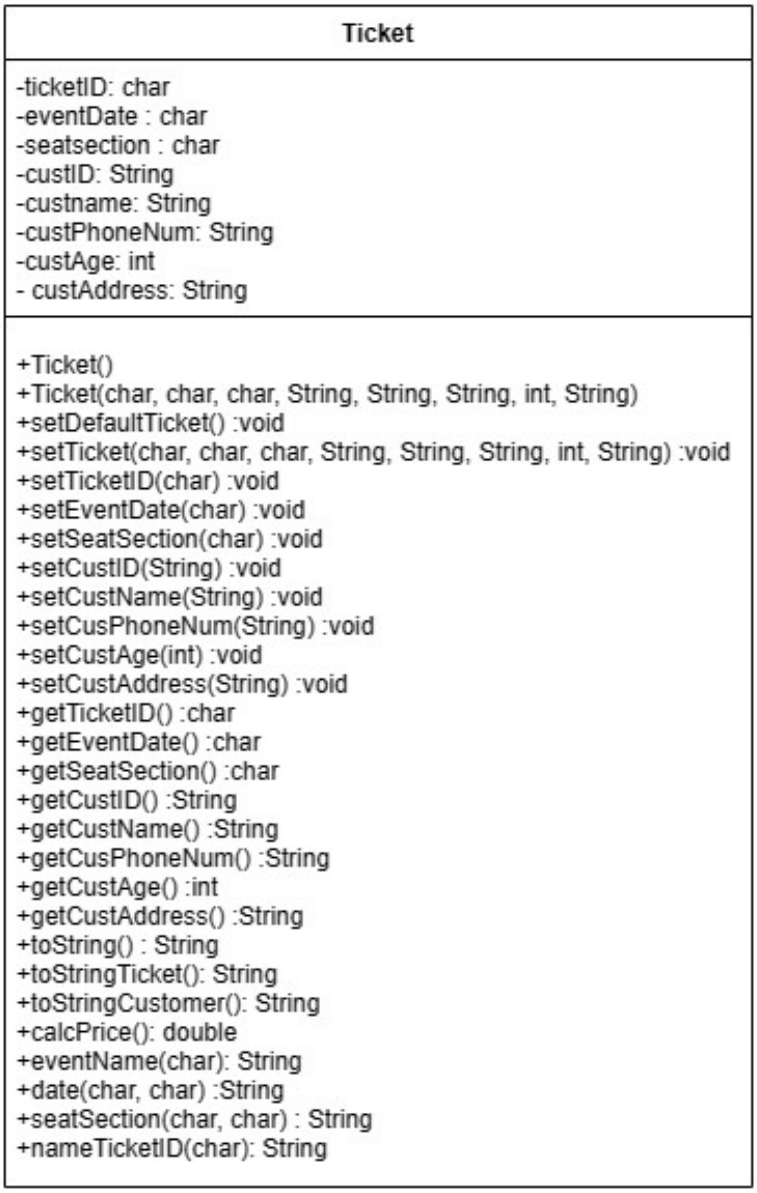
5.1.3 List of Function Prototype

1. `float calcPrice(struct Ticket[] , int );`
2. `void calcTotalSales(struct Ticket[] , int );`
3. `void countSection(struct Ticket[] , int);`
4. `void calcHighest(struct Ticket[]  , int);`
5. `void calcLowest(struct Ticket[]  , int);`
6. `void calcTotalSales(struct Ticket[], int);`
7. `void calcTotalAttendees( truct Ticket[], int);`
8. `void searchData(struct Ticket[], int, int);`
9. `int removeData(struct Ticket[] , int);`

5.2 Java Language

Java is an interpreted language that is in Java, the code is first transformed into bytecode and that bytecode is then executed by the Java Virtual Machine. Java is a high-level language because translation of code takes place into machine language using compiler or interpreter. It only supports a call by value. Java language allows for easy modeling of real-world entities, making it suitable for a wide range of applications.

5.2.1 Class Diagram



4.2.2 List of Array Used

- `Ticket [] t = new Ticket();`

6.0 DIFFERENCE BETWEEN C AND JAVA

There are some difference between C and Java:

C Language	Java Language

• To remove data, we have to overwrite all the current values with the data values of the array next to it with the assist of for loop.
• The final data in the array that is the same with the previous will be removed by deducting the size of array by one.
• Example of coding:
`strcpy(t[s].custID, t[s+1].custID);`
	• To remove a data , we can simply set all the data values to null.
	• Example of coding:
	`t[foundIndex] = new Ticket();`
	`t[foundIndex] = null;`

7.0 CONCLUSION OF FINDING

In our case study, the best programming paradigm best applied for our system is Object-Oriented Paradigm which is Java.

Imperative paradigm coding requires the programmer to code in one source file. This means that all functions including the main function are written in one page and executed sequentially. On the other hand, Object-Oriented Programming (OOP) uses classes where it allows the user to view multiple classes at once if an error occurs or an update is required. Over the course of our case study, we also found that C requires more research, knowledge, and time to implement the coding compared to Java.

Here are the advantages of Java compared to C that we have obtained from our case study:

Object-Oriented (Java)	Paradigm	Imperative (C)
Easier to organize and structure code using OOP principle.	User-friendly	Supports procedural programming but doesn't have native support for OOP which makes the developer need to structure their code using functions and structures
More straightforward syntax compared to C, making it easier for developers to learn and understanding the coding quickly.	Syntax	C has a more complex syntax that might be considered more challenging for beginners.

8.0 SAMPLE INPUT / OUTPUT

8.1 C LANGUAGE

```
Welcome to Ticket Concert Management System!

        ---------------------------------------
                        Ticket List
        ---------------------------------------

        Concert 1:
        Ticket ID: AA001
        Event Name: Aina Abdul Concert
        Event Date Available:
                [1] 24/01/2024 (Wednesday) - 8pm
                [2] 28/02/2024 (Wednesday) - 8pm
        Seat Section:
                [1] CAT 1 (VIP Pass) - RM 300.00
                [2] CAT 2 - RM 250.00
                [3] CAT 3 - RM 100.00

        Concert 2:
        Ticket ID: SN001
        Event Name: Siti Nurhaliza Concert
        Event Date Available:
                [1] 28/09/2024 (Saturday) - 8pm
                [2] 26/10/2024 (Saturday) - 8pm
        Seat Section:
                [1] CAT 1 (VIP Pass) - RM 500.00
                [2] CAT 2 - RM 350.00
                [3] CAT 3 - RM 200.00

        Concert 3:
        Ticket ID: NI001
        Event Name: Noraniza Idris Concert
        Event Date Available:
                [1] 17/05/2024 (Friday) - 8pm
                [2] 14/06/2024 (Friday) - 8pm
        Seat Section:
                [1] CAT 1 (VIP Pass) - RM 150.00
                [2] CAT 2 - RM 100.00
                [3] CAT 3 - RM 50.00
```

```
Customer 1
Customer ID:
C001
Customer Name:
ALI
Customer Phone Number:
010288345
Customer Age:
23
Customer State/City ( Address ):
PAHANG

Ticket Order 1
Choose your preferent Ticket by Entering Concert Code [ 1 - AA001 | 2 - SN001 | 3 - NI001]:
1
Enter Event Date you prefer [1|2]:
2
Enter Seat Section you Prefer [1|2|3]:
3

Customer 2
Customer ID:
C002
Customer Name:
BELLA
Customer Phone Number:
0192773546
Customer Age:
20
Customer State/City ( Address ):
JOHOR

Ticket Order 2
Choose your preferent Ticket by Entering Concert Code [ 1 - AA001 | 2 - SN001 | 3 - NI001]:
2
Enter Event Date you prefer [1|2]:
2
Enter Seat Section you Prefer [1|2|3]:
1

Customer 3
Customer ID:
C003
Customer Name:
ZAIN
Customer Phone Number:
019266382
Customer Age:
34
Customer State/City ( Address ):
KELANTAN

Ticket Order 3
Choose your preferent Ticket by Entering Concert Code [ 1 - AA001 | 2 - SN001 | 3 - NI001]:
2
Enter Event Date you prefer [1|2]:
1
Enter Seat Section you Prefer [1|2|3]:
1
```

```
Customer 4
Customer ID:
C004
Customer Name:
DINA
Customer Phone Number:
0102966259
Customer Age:
20
Customer State/City ( Address ):
KL

Ticket Order 4
Choose your preferent Ticket by Entering Concert Code [ 1 - AA001 | 2 - SN001 | 3 - NI001]:
1
Enter Event Date you prefer [1|2]:
1
Enter Seat Section you Prefer [1|2|3]:
1


Customer 5
Customer ID:
C005
Customer Name:
FARAH
Customer Phone Number:
0167388324
Customer Age:
34
Customer State/City ( Address ):
PERAK

Ticket Order 5
Choose your preferent Ticket by Entering Concert Code [ 1 - AA001 | 2 - SN001 | 3 - NI001]:
3
Enter Event Date you prefer [1|2]:
2
Enter Seat Section you Prefer [1|2|3]:
1

Total Earnings for Each Concert:
        Aina Abdul's Concert : RM400.00
        Siti Nurhaliza's Concert : RM1000.00
        Noraniza Idris's Concert : RM150.00


Total Attendees for Each Concert:
        Aina Abdul Concert : 2
        Siti Nurhaliza Concert : 2
        Noraniza Idris : 1


Name of highest attended event : Aina Abdul's Concert
Total attendees for the highest event : 2
Name of highest sales gained from an event : Siti Nurhaliza's' Concert
Total sales for the highest event : 1000.00

Name of lowest attended event : Noraniza Idris's Concert
Total attendees for the lowest event : 1
Name of lowest sales gained from an event : Noraniza Idris's Concert
Total sales for the lowest event : 150.00

UPDATE PHONE NUMBER
Enter [Y] to continue update phone number or [N] to skip this process:
```

```
UPDATE PHONE NUMBER
Enter [Y] to continue update phone number or [N] to skip this process:
Y

Enter customer ID to update phone number:
C001
Customer found!

Enter the new phone number to update:
0182555637

Phone number has been updated.

UPDATED DETAILS
Customer ID: C001
Customer Name: ALI
Phone Number: 0182555637
Age: 23
Address: PAHANG


Enter [Y] to continue update phone number or [N] to skip this process:
N
```

```
===== Customer Data 1 =====

Event Name: Aina Abdul Concert
Event Date: 28/02/2024 (Wednesday) - 8pm
Seat Section: CAT 3
Price: 100.00

Customer ID: C001
Customer Name: ALI
Phone Number: 0182555637
Age: 23
Address: PAHANG


===== Customer Data 2 =====

Event Name: Siti Nurhaliza Concert
Event Date: 28/01/2024 (Saturday) - 8pm
Seat Section: CAT 1 (VIP Pass)
Price: 500.00

Customer ID: C003
Customer Name: ZAIN
Phone Number: 019266382
Age: 34
Address: KELANTAN
```

```
===== Customer Data 3 =====

Event Name: Aina Abdul Concert
Event Date: 24/01/2024 (Wednesday) - 8pm
Seat Section: CAT 1 (VIP Pass)
Price: 300.00

Customer ID: C004
Customer Name: DINA
Phone Number: 0102966259
Age: 20
Address: KL


===== Customer Data 4 =====

Event Name: Noraniza Idris Concert
Event Date: 14/02/2024 (Friday) - 8pm
Seat Section: CAT 1 (VIP Pass)
Price: 150.00

Customer ID: C005
Customer Name: FARAH
Phone Number: 0167388324
Age: 34
Address: PERAK



Thank You for Using Our System!
--------------------------------
Process exited after 676.7 seconds with return value 0
Press any key to continue . . .
```

8.2 JAVA LANGUAGE

```
Welcome to Ticket Concert Management System!

        ----------------------------------------
                        Ticket List
        ----------------------------------------

        Concert 1:
        Ticket ID: AA001
        Event Name: Aina Abdul Concert
        Event Date Available:
                [1] 24/01/2024 (Wednesday) - 8pm
                [2] 28/02/2024 (Wednesday) - 8pm
        Seat Section:
                [1] CAT 1 (VIP Pass) - RM 300.00
                [2] CAT 2 - RM 250.00
                [3] CAT 3 - RM 100.00

        Concert 2:
        Ticket ID: SN001
        Event Name: Siti Nurhaliza Concert
        Event Date Available:
                [1] 28/09/2024 (Saturday) - 8pm
                [2] 26/10/2024 (Saturday) - 8pm
        Seat Section:
                [1] CAT 1 (VIP Pass) - RM 500.00
                [2] CAT 2 - RM 350.00
                [3] CAT 3 - RM 200.00

        Concert 3:
        Ticket ID: NI001
        Event Name: Noraniza Idris Concert
        Event Date Available:
                [1]17/05/2024 (Friday) - 8pm
                [2]14/06/2024 (Friday) - 8pm
        Seat Section:
                [1] CAT 1 (VIP Pass) - RM 150.00
                [2] CAT 2 - RM 100.00
                [3] CAT 3 - RM 50.00
```

```
Customer 1
Customer ID:
C001
Customer Name:
ALI
Customer Phone Number:
010288345
Customer Age:
23
Customer State/City ( Address ):
PAHANG

Ticket Order 1 :
Choose your preferent Ticket by Entering Concert Code [1 - AA001|2 - SN001|3 - NI001]:
1
Enter Event Date you Prefer [1|2]:
2
Enter Seat Section you Prefer [1|2|3]:
3
Customer 2
Customer ID:
C002
Customer Name:
BELLA
Customer Phone Number:
0192773546
Customer Age:
20
Customer State/City ( Address ):
JOHOR

Ticket Order 2 :
Choose your preferent Ticket by Entering Concert Code [1 - AA001|2 - SN001|3 - NI001]:
2
Enter Event Date you Prefer [1|2]:
2
Enter Seat Section you Prefer [1|2|3]:
1
```

```
Customer 3
Customer ID:
C003
Customer Name:
ZAIN
Customer Phone Number:
019266382
Customer Age:
34
Customer State/City ( Address ):
KELANTAN

Ticket Order 3 :
Choose your preferent Ticket by Entering Concert Code [1 - AA001|2 - SN001|3 - NI001]:
2
Enter Event Date you Prefer [1|2]:
1
Enter Seat Section you Prefer [1|2|3]:
1
```

```
Customer 4
Customer ID:
C004
Customer Name:
DINA
Customer Phone Number:
0102966259
Customer Age:
20
Customer State/City ( Address ):
KL

Ticket Order 4 :
Choose your preferent Ticket by Entering Concert Code [1 - AA001|2 - SN001|3 - NI001]:
1
Enter Event Date you Prefer [1|2]:
1
Enter Seat Section you Prefer [1|2|3]:
1

Customer 5
Customer ID:
C005
Customer Name:
FARAH
Customer Phone Number:
0167388324
Customer Age:
34
Customer State/City ( Address ):
PERAK

Ticket Order 5 :
Choose your preferent Ticket by Entering Concert Code [1 - AA001|2 - SN001|3 - NI001]:
3
Enter Event Date you Prefer [1|2]:
2
Enter Seat Section you Prefer [1|2|3]:
1

Total Earnings for Each Concert:
        Aina Abdul's Concert: RM 400.0
        Siti Nurhaliza's Concert: RM 1000.0
        Noraniza Idris's Concert: RM 150.0

Total Attendees for Each Concert:
        Aina Abdul's Concert: 2
        Siti Nurhaliza's Concert: 2
        Noraniza Idris's Concert: 1

Name of highest attended event: Aina Abdul's Concert
Total attendees for the highest event: 2
Name of highest sales gained from an event: Siti Nurhaliza's Concert
Total sales for highest sales event: RM 1000.0

Name of lowest attended event: Noraniza Idris's Concert
Total attendees for the lowest event: 1
Name of lowest sales gained from an event: Noraniza Idris's Concert
Total sales for lowest sales event: RM 150.0

UPDATE PHONE NUMBER

Enter [Y] to continue update phone number or [N] to skip this process:
```

```
UPDATE PHONE NUMBER

Enter [Y] to continue update phone number or [N] to skip this process:
Y

Enter customer ID to update phone number:
C001
Customer found!

Enter the new phone number to update:
0182555637

Phone number has been updated.

UPDATED DETAILS
Customer ID: C001
Customer Name: ALI
Phone Number: 0182555637
Age: 23
Address: PAHANG

Enter [Y] to continue update phone number or [N] to skip this process:
N

Enter [Y] to continue update phone number or [N] to skip this process:
N

REMOVE CUSTOMER ORDER
Enter [Y] to continue or [N] to skip this process:
Y
Enter Customer ID that want to remove:
C002
Customer found!
Deleting...

Customer data for C002 is successfully delete.

Enter [Y] to continue or [N] to skip this process:
N


Customer's Ticket Information:

===== Customer 1 ======

Ticket ID: AA001
Event Name: Aina Abdul's Concert
Event Date: 28/02/2024 (Wednesday) - 8pm
Seat Section: CAT 3 - RM 100.00

Customer ID: C001
Customer Name: ALI
Phone Number: 0182555639
Age: 23
Address: PAHANG
```

```
===== Customer 2 ======

Ticket ID: SN001
Event Name: Siti Nurhaliza's Concert
Event Date: 28/09/2024 (Saturday) - 8pm
Seat Section: CAT 1 - (VIP Pass) - RM 500.00

Customer ID: C003
Customer Name: ZAIN
Phone Number: 019266382
Age: 34
Address: KELANTAN
===== Customer 3 ======

Ticket ID: AA001
Event Name: Aina Abdul's Concert
Event Date: 24/01/2024 (Wednesday) - 8pm
Seat Section: CAT 1 - (VIP Pass) - RM 300.00

Customer ID: C004
Customer Name: DINA
Phone Number: 0102966259
Age: 20
Address: KL

===== Customer 4 ======

Ticket ID: NI001
Event Name: Noraniza Idris's Concert
Event Date: 14/06/2024 (Friday) - 8pm
Seat Section: CAT 1 - (VIP Pass) - RM 150.00

Customer ID: C005
Customer Name: FARAH
Phone Number: 0167388324
Age: 34
Address: PERAK


Thank You for Using Our System!
```