School of Computing

Faculty of Engineering

UNIVERSITI TEKNOLOGI MALAYSIA

DATA STRUCTURE & ALGORITHM

(SECJ2013)

SEMESTER 1 2020/2021

Mini PROJECT Documentation

SPORT CENTRE REGISTRATION SYSTEM

By

Arif Amiruddin bin Sadiran (000503-10-1007) ***Leader***

Muhammad ‘Afif Azhan bin Mohd Ismail (000809-05-0265)

Muhammad Iskandar Zulqarnain bin Mohd Ishak (001108-01-0679)

SECTION 03

Lecturer

Ms. Lizawati binti Mi Yusuf

Date

30 January 2021

Table of Content

PART 1: INTRODUCTION

1.1 Synopsis Project

For our final project of SECJ2013 for this semester, we are required to make a program that works like a sport centre system. We are required to apply data structures that we had learnt in our class like linked list, queue, stack and tree.

For this project, we had applied two types of data structure which are, linked list and queue. We used the linked list to create a list of customers that had attended the sport centre, while queue is used to generate a queue of equipment that can be rented to the customers. For our queue, we had used two queues where one for the available equipment for rent, and the second one for rented equipment of the day. We used queues instead of stacks because, if not all equipment is rented on the specific day, the system will insert yesterday's rented equipment into the available equipment queue.

So, if a customer wants to rent an equipment, they will only be able to rent the equipment that has not been rented since yesterday. This allows all equipment to be used and not stored in the storage forever.

Our system has six functions or algorithms. First, it can display the lists of customers that had attended the sport centre in the past. Other than that, it allows the user of the system to insert a customer's details into the back, front or in the middle of the list.

The third function is sorting the list according to the customers' name, IC and age. The system can sort the list in either ascending or descending order using the insertion sort method. Not only that, the system also allows data searching to search for the details of a certain customer in the list. The user can choose to search by name, by ICs or by month of attendance. If the inserted detail matches, it will display the details of the customer(s).The fifth one is data deletion function. This allows the user to delete a chosen customer from the list. Users only need to enter a number based on the displayed list to delete the respective customer.

Lastly, the last function is the daily equipment rental. This allows customers to rent equipment from the sports centre for the specific day. If there are any equipment that are available for rent, the customer can insert their details and a receipt will be generated. Users need to show the receipt to the rent counter to retrieve their item. Users can also check the list of rented items in the past. This algorithm applies the queue data structure where the equipment list will be in a queue. This allows the equipment to be in First In First Out movement, to allow all available equipment to be rented before starting a new rent cycle

1.2 Objective of The Project

The objectives of the project are :-

- To develop deeper understanding on the DSA subject.
- Gives ideas on how to implement DSA concepts in a real life situation.
- Develop team working and soft skills among team members.
- To develop a sport center system with insert, delete, display, sort, search and rent equipment.

PART 2: SYSTEM ANALYSIS AND DESIGN (USE CASE, FLOWCHART AND CLASS DIAGRAM)

In this section, we identified the requirements and the design of the system. We also provide the use case diagram, flowchart and class diagram for every module in our sports centre system.

2.1 System Requirements

Use Case Diagram

Example:

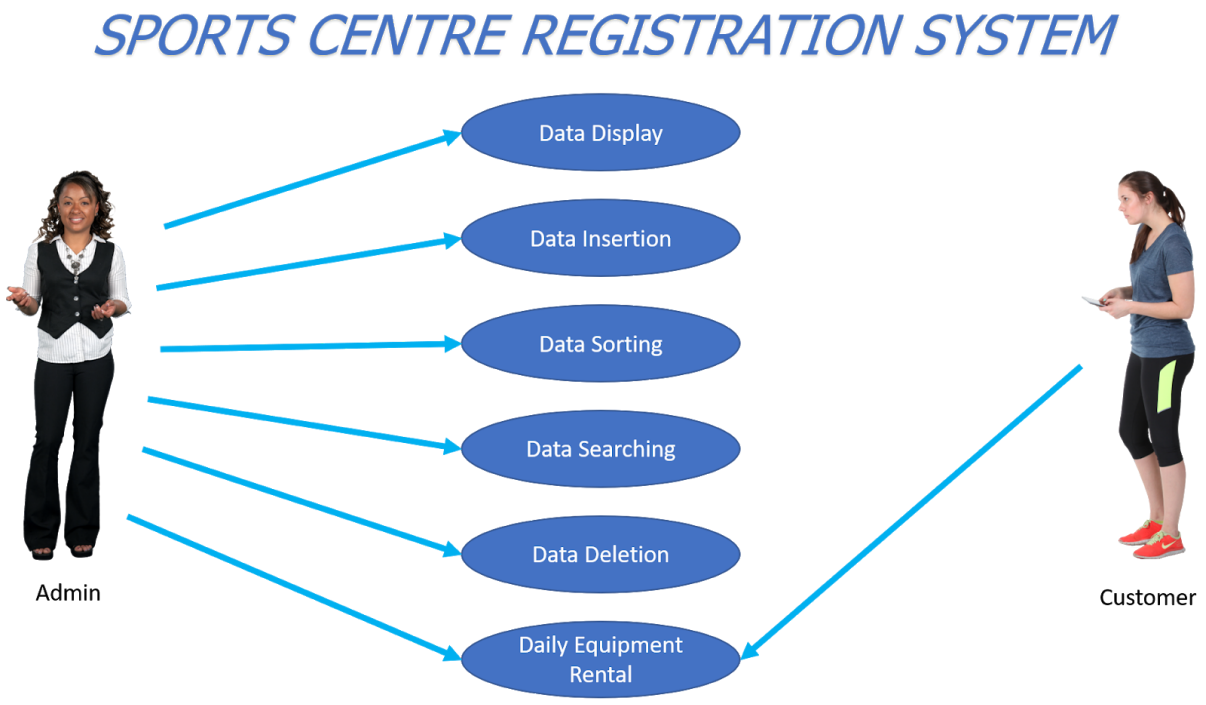


Figure 1: Use Case Diagram for Sports Centre Registration System

Use Case Description for Sports Centre Registration System

The system users are admin and customer.

Actor	Task
Admin	Admin will perform all functions in the system. The functions include displaying data of current customers, inserting new data for customers who want to book the sports centre, sorting data within certain criteria, searching data within certain criteria, deleting data of customers who already finish using the sports facilities and performing daily equipment rental process for customers who want to rent sports equipment.
Customer	Customers may access to look at available equipment including futsal ball and racket. The customer could also check the rental record and the equipment's status wherever it is still available or not.

Detail Description for Each Use Cases

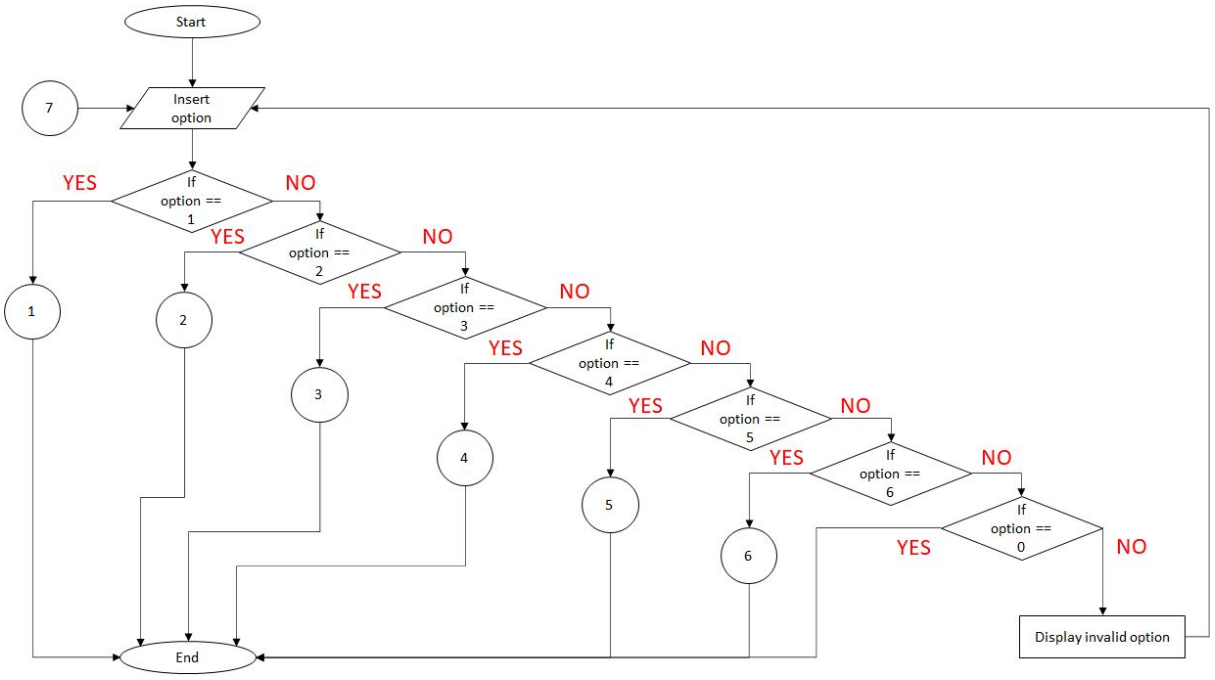
The system has 6 main use cases.

Use Case	Task
Data Display	Display current customer list in the system that already registered to use the sport centre (list of customers already assigned in the program's codes)
Data Insertion	Insert new customers' details to the sports centre registration system who wants to book the facilities. Admin may have several choices to input customers' details via front, middle or at the end of the list.
Data Sorting	Sort customers' details through demanded criteria such as name, identification card number and age.
Data Searching	Search customers' data through demanded criteria such as name, identification card number and month of booking.
Data Deletion	Delete the customers' data when their session ended after using the sport centre facilities.
Daily Equipment Rental	Provide daily equipment rent system to allow customers to rent some sports equipment such as futsal ball, racquet and swimsuit. This function also allows the admin to display the list of rented equipment.

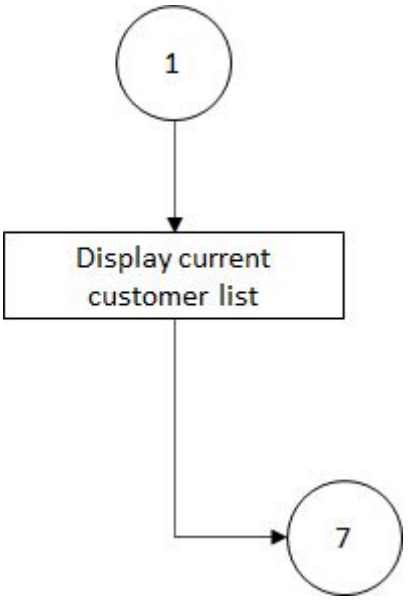
2.2 System Design

Algorithm: Flowchart for each module.

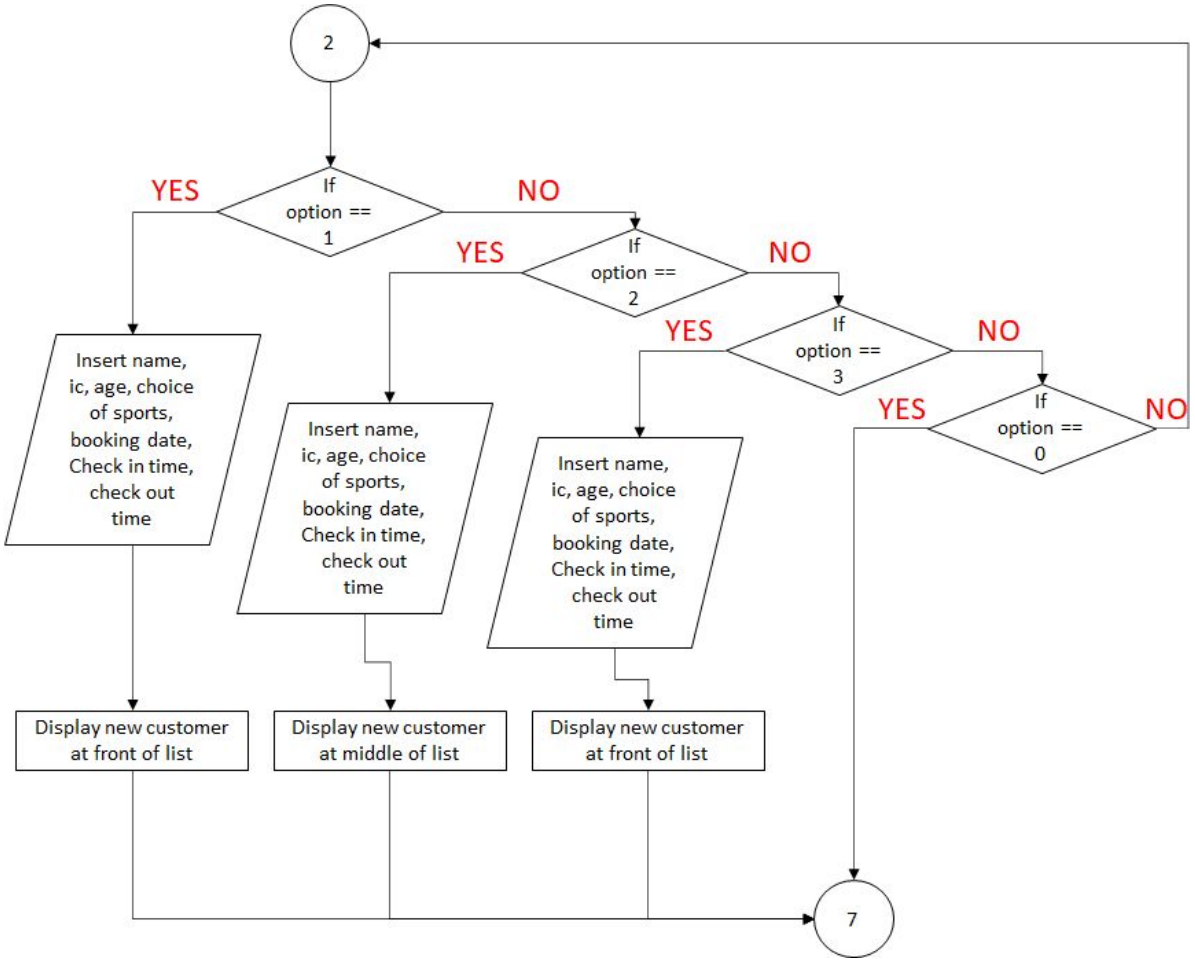
Flowchart 0: Overall System Overview
Prepared By;.Iskandar Zulqarnain Mohd Ishak



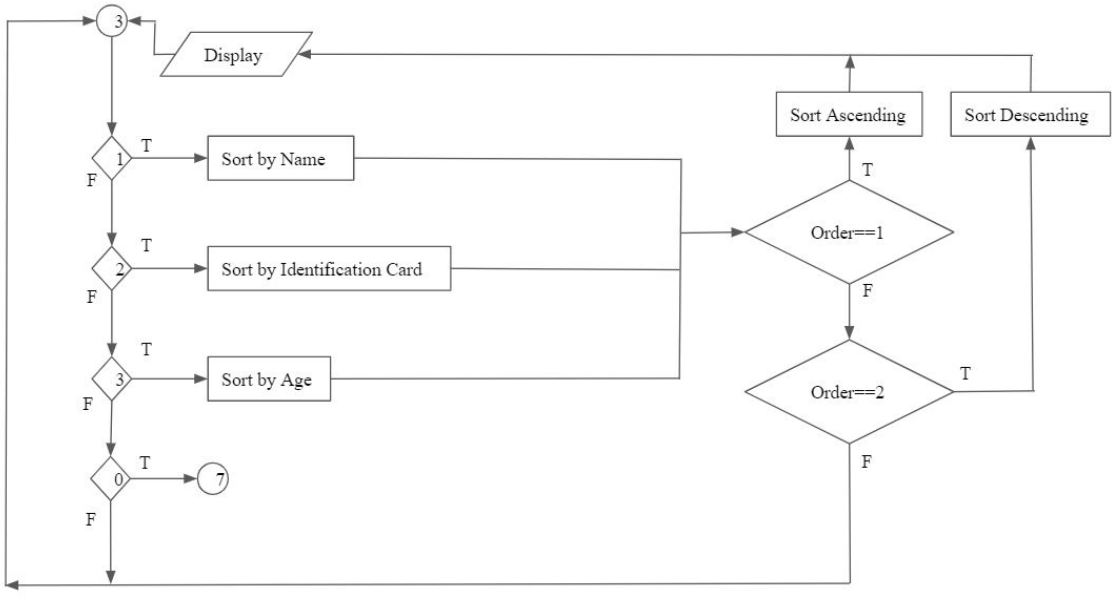
Flowchart 1: Data Display
Prepared By;.Iskandar Zulqarnain Mohd Ishak



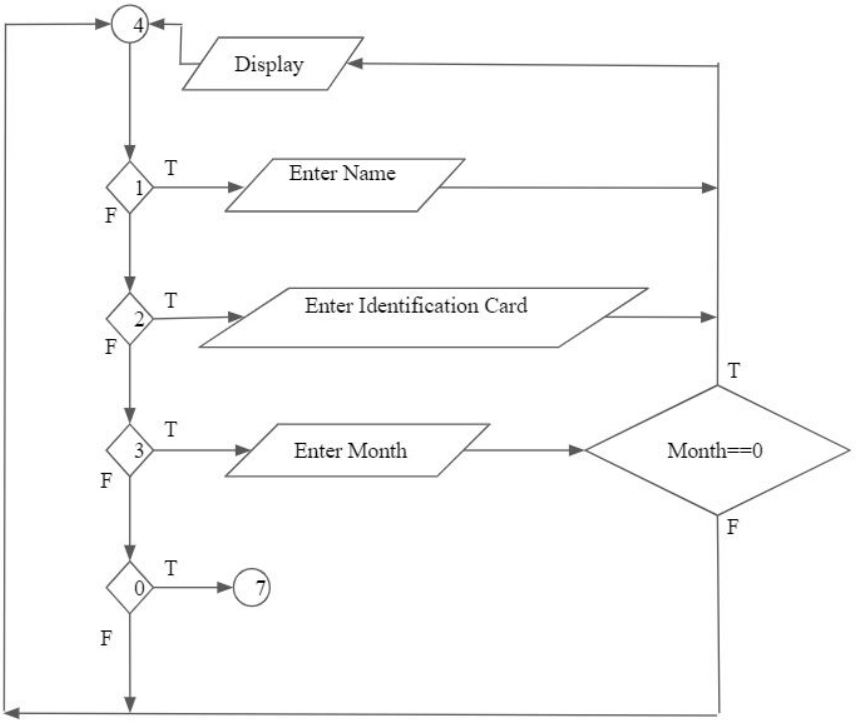
Flowchart 2: Data Insertion
Prepared By; Iskandar Zulqarnain Mohd Ishak



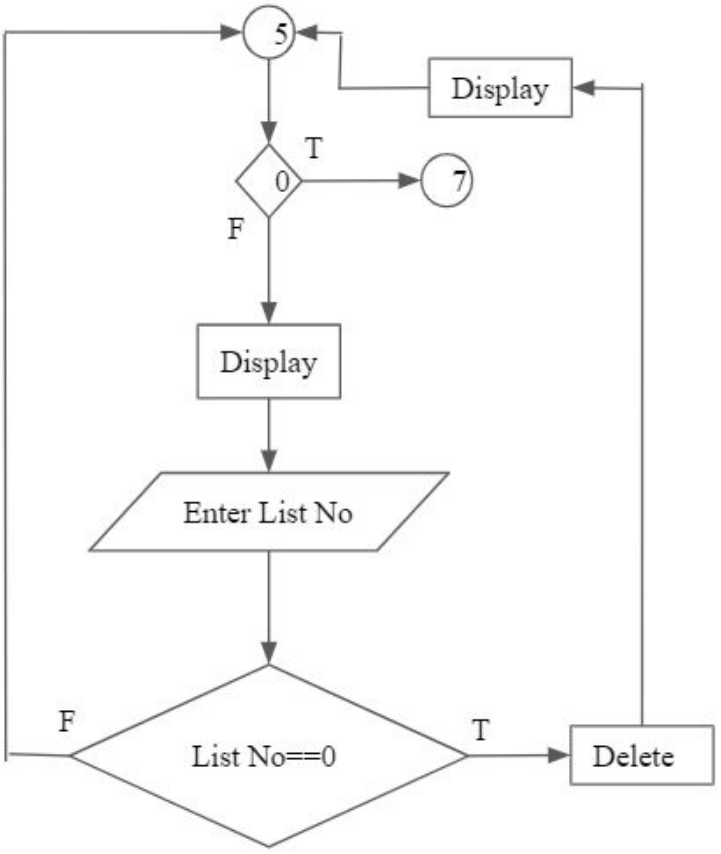
Flowchart 3: Data Sorting
Prepared By; Arif Amiruddin bin Sadiran



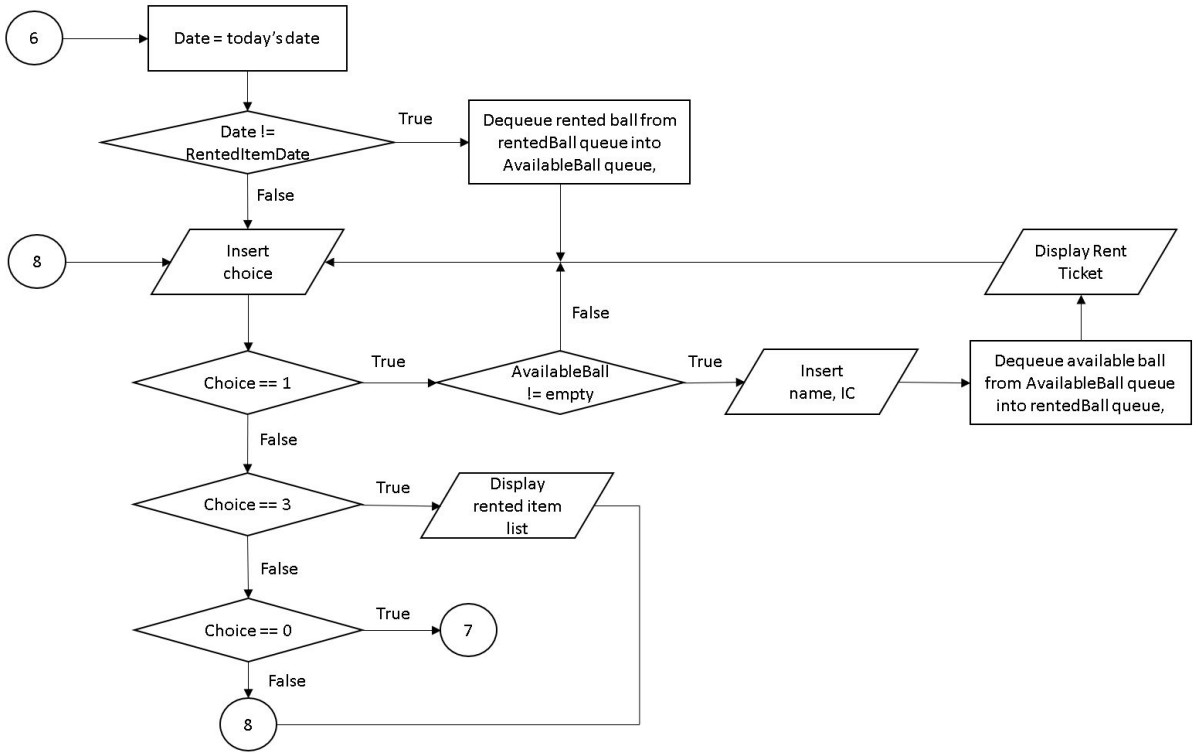
Flowchart 4: Data Searching
Prepared By; Arif Amiruddin bin Sadiran



Flowchart 5: Data Deletion
Prepared By; Arif Amiruddin bin Sadiran



Flowchart 6: Daily Equipment Rental
Prepared By; ‘Afif Azhan Mohd Ismail



PART 3: SYSTEM PROTOTYPE

Main Menu

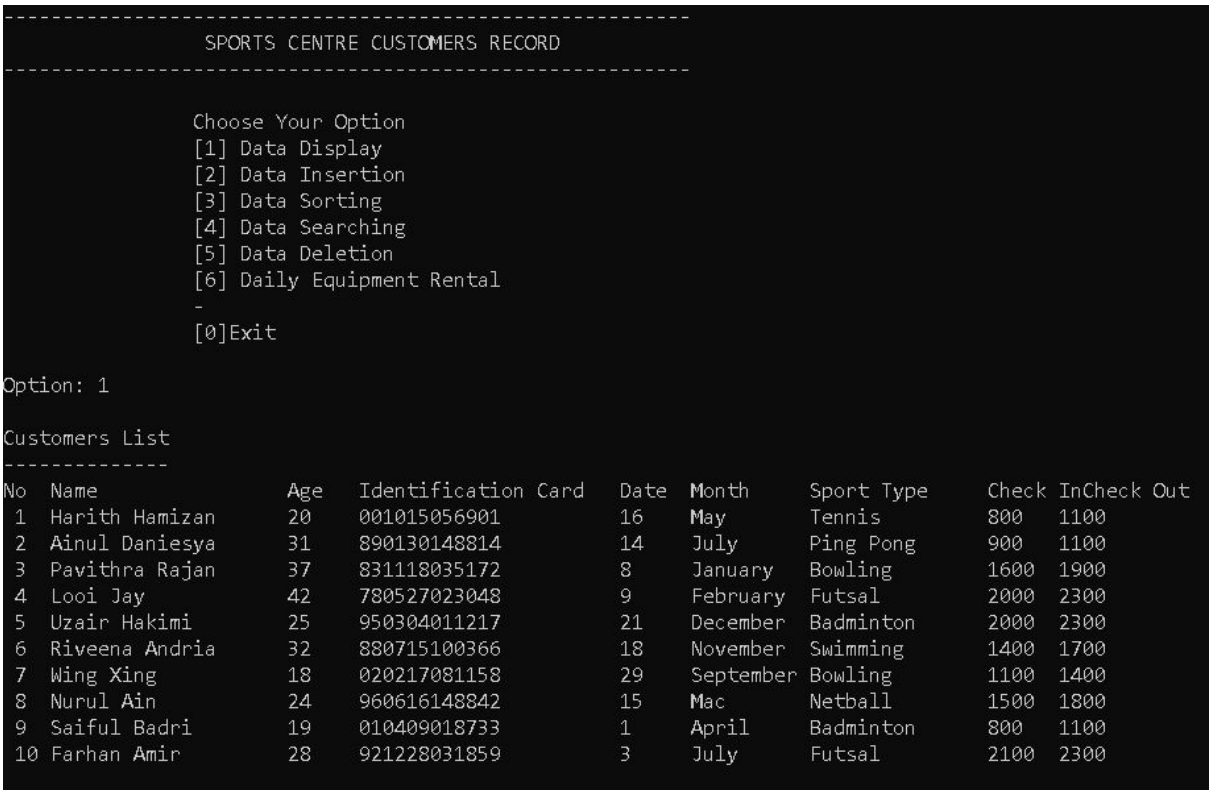
This screen shows the main menu of the system. Users must enter an integer between 1-6 to enter a specific module. To exit the program, the user needs to enter '0'. If the user enters an integer other than 0-6, the program will display an error message and the screen will be displayed again. Then, users are allowed to enter their input again.

This menu will be displayed if the user chose to exit the program.

Explanation Prepared By: 'Afif Azhan Mohd Ismail

Code Prepared By: Iskandar Zulqarnain Mohd Ishak

Module 1 - Data Display



```
------------------------------------------------------------
                SPORTS CENTRE CUSTOMERS RECORD
------------------------------------------------------------

               Choose Your Option
               [1] Data Display
               [2] Data Insertion
               [3] Data Sorting
               [4] Data Searching
               [5] Data Deletion
               [6] Daily Equipment Rental
               -
               [0]Exit

Option: 1

Customers List
--------------
No  Name              Age   Identification Card   Date  Month     Sport Type     Check InCheck Out
 1  Harith Hamizan    20    001015056901          16    May       Tennis         800   1100
 2  Ainul Daniesya    31    890130148814          14    July      Ping Pong      900   1100
 3  Pavithra Rajan    37    831118035172          8     January   Bowling        1600  1900
 4  Looi Jay          42    780527023048          9     February  Futsal         2000  2300
 5  Uzair Hakimi      25    950304011217          21    December  Badminton      2000  2300
 6  Riveena Andria    32    880715100366          18    November  Swimming       1400  1700
 7  Wing Xing         18    020217081158          29    September Bowling        1100  1400
 8  Nurul Ain         24    960616148842          15    Mac       Netball        1500  1800
 9  Saiful Badri      19    010409018733          1     April     Badminton      800   1100
 10 Farhan Amir       28    921228031859          3     July      Futsal         2100  2300
```

Screen 2: Book Searching System

This screen shows the list of customers in the system record. The list is shown below the main menu if the user inserted ‘1’ in the main menu option. If there are no customers in the record, nothing will be displayed except the system telling you that the list is empty. To return to the main menu, the user needs to click any key on the keyboard.

Explanation Prepared By: ‘Afif Azhan Mohd Ismail

Code Prepared By: Iskandar Zulqarnain Mohd Ishak

Module 2 - Data Insertion

```
------------------------------------------------------------
                    DATA INSERTION
------------------------------------------------------------

Customers List
--------------
No  Name               Age   Identification Card   Date  Month     Sport Type     Check InCheck Out
 1  Ahmad Ali          17    030801060298          9     July      Badminton      1800  2000
 2  Harith Hamizan     20    001015056901          16    May       Tennis         800   1100
 3  Ainul Daniesya     31    890130148814          14    July      Ping Pong      900   1100
 4  Pavithra Rajan     37    831118035172          8     January   Bowling        1600  1900
 5  Looi Jay           42    780527023048          9     February  Futsal         2000  2300
 6  Uzair Hakimi       25    950304011217          21    December  Badminton      2000  2300
 7  Riveena Andria     32    880715100366          18    November  Swimming       1400  1700
 8  Wing Xing          18    020217081158          29    September Bowling        1100  1400
 9  Nurul Ain          24    960616148842          15    Mac       Netball        1500  1800
 10 Saiful Badri       19    010409018733          1     April     Badminton      800   1100
 11 Farhan Amir        28    921228031859          3     July      Futsal         2100  2300


               Choose Your Option
               [1] Insert data in the beginning of the list
               [2] Insert data in the middle of the list
               [3] Insert data in the end of the list
               -
               [0]Back

Option:
```

This screen shows the data insertion function in our program. The user can choose either to insert a new customer record into the either front, middle or the end of the list by entering an integer between 1-3. If the user inserted 0, they will be directed back to the main menu. If they inserted other than that, the program will prompt an error message and the screen will be displayed again.

```
------------------------------------------------------------
                    DATA INSERTION
------------------------------------------------------------

Customers List
--------------
No  Name               Age   Identification Card   Date  Month     Sport Type     Check InCheck Out
 1  Harith Hamizan     20    001015056901          16    May       Tennis         800   1100
 2  Ainul Daniesya     31    890130148814          14    July      Ping Pong      900   1100
 3  Pavithra Rajan     37    831118035172          8     January   Bowling        1600  1900
 4  Looi Jay           42    780527023048          9     February  Futsal         2000  2300
 5  Uzair Hakimi       25    950304011217          21    December  Badminton      2000  2300
 6  Riveena Andria     32    880715100366          18    November  Swimming       1400  1700
 7  Wing Xing          18    020217081158          29    September Bowling        1100  1400
 8  Nurul Ain          24    960616148842          15    Mac       Netball        1500  1800
 9  Saiful Badri       19    010409018733          1     April     Badminton      800   1100
 10 Farhan Amir        28    921228031859          3     July      Futsal         2100  2300


               Choose Your Option
               [1] Insert data in the beginning of the list
               [2] Insert data in the middle of the list
               [3] Insert data in the end of the list
               -
               [0]Back

Option: 4

Invalid option! Please try again.
```

This screen shows the invalid option message whenever the user inserted a wrong input.

```
               Choose Your Option
               [1] Insert data in the beginning of the list
               [2] Insert data in the middle of the list
               [3] Insert data in the end of the list
               -
               [0]Back

Option: 1

Please fill the information for the new data :

Name : Ahmad Ali
Age  : 17
IC   : 030801060298
Sport Type  : [1] Futsal
              [2] Badminton
              [3] Netball
              [4] Bowling
              [5] Swimming
              [6] Ping Pong
              [7] Tennis

              Choice: 2

Date of Check In and Check Out (i.e.: 7 January) : 9 July
Check In time (24H)  : 1800
Check Out time (24H) : 2000
```

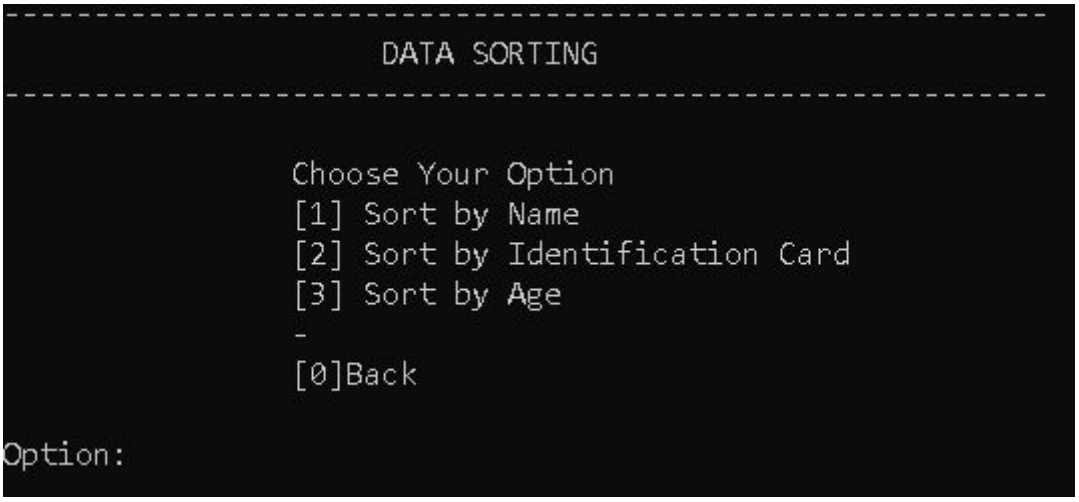
The next screen shows the output when the user chose to insert new data into the list. The program will ask the user to insert the details of the customer which are the name, age, identification ID, the sport that they are playing, the date of attendance, the check in time and the check out time. This output is the same for every 1-3 choices.

```
The list after insertion process.

Customers List
--------------
No  Name              Age   Identification Card   Date  Month     Sport Type     Check InCheck Out
 1  Ahmad Ali         17    030801060298          9     July      Badminton      1800  2000
 2  Harith Hamizan    20    001015056901          16    May       Tennis         800   1100
 3  Ainul Daniesya    31    890130148814          14    July      Ping Pong      900   1100
 4  Pavithra Rajan    37    831118035172          8     January   Bowling        1600  1900
 5  Looi Jay          42    780527023048          9     February  Futsal         2000  2300
 6  Uzair Hakimi      25    950304011217          21    December  Badminton      2000  2300
 7  Riveena Andria    32    880715100366          18    November  Swimming       1400  1700
 8  Wing Xing         18    020217081158          29    September Bowling        1100  1400
 9  Nurul Ain         24    960616148842          15    Mac       Netball        1500  1800
 10 Saiful Badri      19    010409018733          1     April     Badminton      800   1100
 11 Farhan Amir       28    921228031859          3     July      Futsal         2100  2300

Returning to main menu.
```

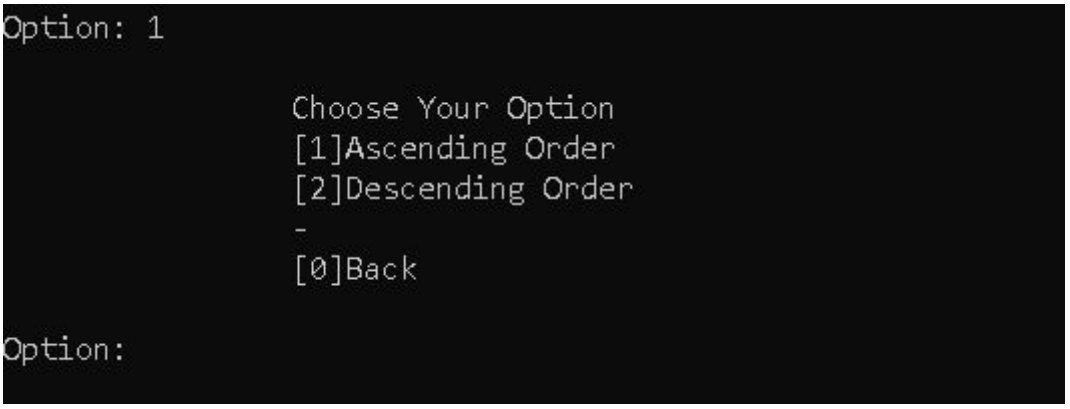
After the user inserted the details of the new customer into the list, the program will display the new list after the insertion process. Based on the first figure of this module, the user chose to enter the new data at the front of the list. So, we can say here that the customer "Ahmad Ali" has been added at the front of the list. After that, user will be directed to the main menu.

Explanation Prepared By: 'Afif Azhan Mohd Ismail
Code Prepared By: Iskandar Zulqarnain Mohd Ishak

Module 3 - Data Sorting



This screen shows the first menu of the data sorting function for the program. The user needs to choose either to sort the list either by name, identification number or by their age by entering integer between 1-3. To return to the main menu, the user needs to insert the integer 0. Inserting an integer other than 0-3 will produce an invalid option message. The screen will be displayed again so the user can choose a new option.



Once the user inserted his option, they would need to choose either to sort the list in ascending or descending order. This choice will be displayed the same for the sort by IC and age function. If the user inserted an integer that is not in the 0-2 range, invalid input message will be displayed and the user will be directed back to the sorting function main menu.

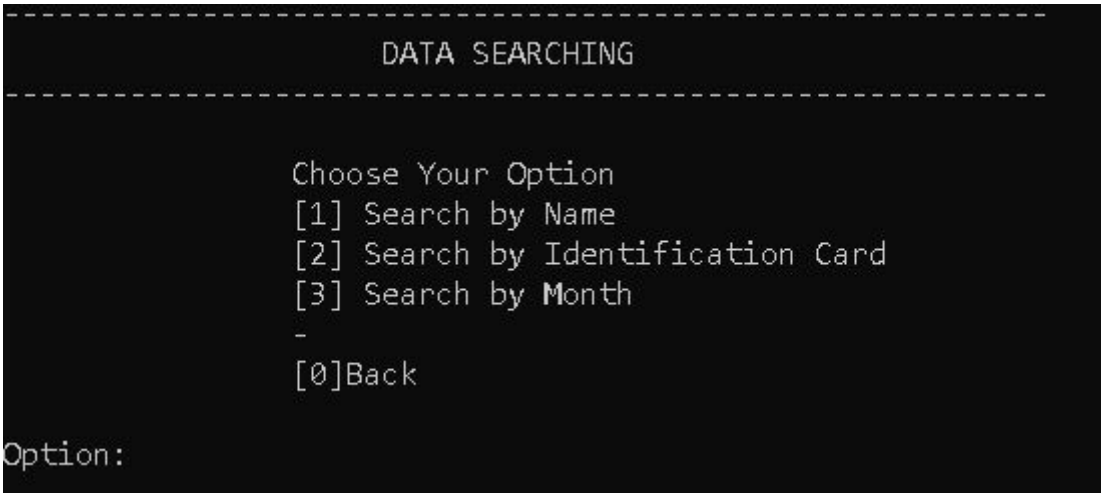
```
               Choose Your Option
               [1]Ascending Order
               [2]Descending Order
               -
               [0]Back

Option: 1

Sorting process successful. Please refer to the new list below.

Customers List
--------------
No  Name              Age   Identification Card   Date  Month     Sport Type     Check InCheck Out
 1  Ainul Daniesya    31    890130148814          14    July      Ping Pong      900   1100
 2  Farhan Amir       28    921228031859          3     July      Futsal         2100  2300
 3  Harith Hamizan    20    001015056901          16    May       Tennis         800   1100
 4  Looi Jay          42    780527023048          9     February  Futsal         2000  2300
 5  Nurul Ain         24    960616148842          15    Mac       Netball        1500  1800
 6  Pavithra Rajan    37    831118035172          8     January   Bowling        1600  1900
 7  Riveena Andria    32    880715100366          18    November  Swimming       1400  1700
 8  Saiful Badri      19    010409018733          1     April     Badminton      800   1100
 9  Uzair Hakimi      25    950304011217          21    December  Badminton      2000  2300
 10 Wing Xing         18    020217081158          29    September Bowling        1100  1400
```

If the sorting process is successful. The program will display the new list after sorting depending on the user's choice. For this example, the user chose to sort the list in ascending order based on the customer's name. We can see that the number 1 in the list started with 'A' followed by 'F' in the second one. After displaying the new list, the program will display the main menu of the sorting function.

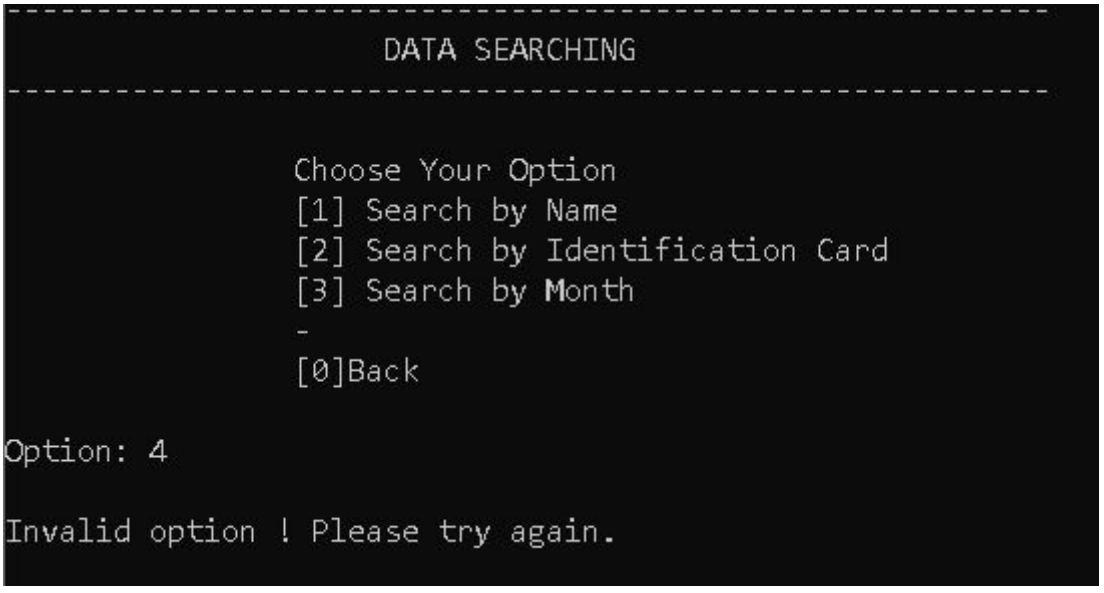
Explanation Prepared By: 'Afif Azhan Mohd Ismail

Code Prepared By: Arif Amiruddin bin Sadiran and Iskandar Zulqarnain Mohd Ishak

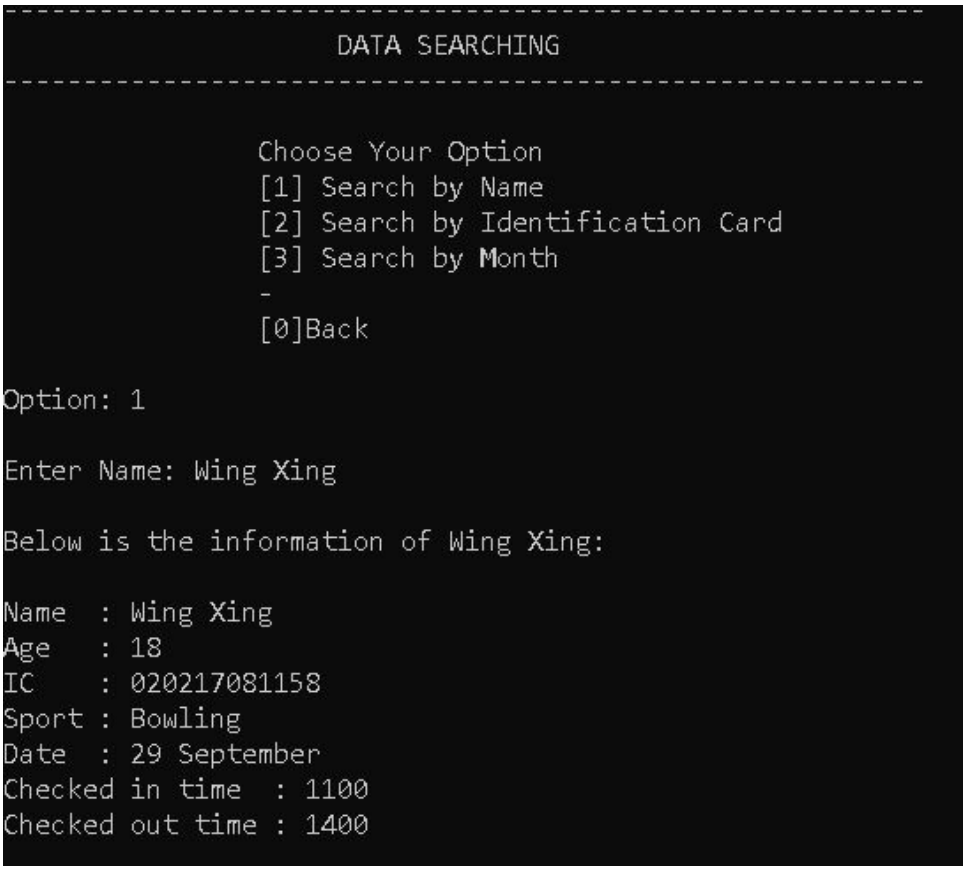
Module 4 - Data Searching



This screen shows the main menu of the searching function where users need to enter an integer between 0 - 3. To use the searching function either based on search by name, by identification number or by month of attendance, they need to enter 1, 2, and 3 respectively. To return to the main menu, the user will need to enter 0. If the user entered an integer other than that, the program will prompt an error message and the screen will be displayed again.



Based on the figure above, we can see that the program displayed an error message if the user inserted 4, because the valid integer is in the range of 0-3.



For this screen, the user had chosen to search a customer by their name. He needs to enter the full name of the customer and the program will display the customer's details.

The figure above shows the display whenever the name that the user entered does not match with any of the customers in the list. By clicking any key on the keyboard, the user will be redirected to the searching function main menu.

```
Option: 2

Enter Identification Card: 001015056901

Below is the information about the customer with IC 001015056901

Name  : Harith Hamizan
Age   : 20
IC    : 001015056901
Sport : Tennis
Date  : 16 May
Checked in time  : 800
Checked out time : 1100
```

Other than that, the figure above shows the display when the user wanted to search a customer's details by using their IC. The program will display the same as the search by name function.

```
Option: 2

Enter Identification Card: 780527023041

There is no customer with IC 780527023041 in the record.
```

This screen shows the result when the inserted IC does not match any of the customer's IC in the list. The user will be directed back to the searching function main menu.

```
Option: 3
                Choose which month
                [1] January
                [2] February
                [3] March
                [4] April
                [5] May
                [6] June
                [7] July
                [8] August
                [9] September
                [10] October
                [11] November
                [12] December
                -
                [0] Back

Option: 7

Below is the customer(s) that used the sport centre in July :

No  Name              Age   Identification Card   Date  Month     Sport Type     Check In  Check Out
1   Ainul Daniesya    31    890130148814          14    July      Ping Pong      900       1100
2   Farhan Amir       28    921228031859          3     July      Futsal         2100      2300
```

For the 3rd option of the searching function, the user needs to choose the month's customers. Like the example above, the user wanted to see the details of the customers that had attended the sport centre in July. The program will display a list of customers instead of a single person. This is because it is impossible for a customer to have the same IC number plus it is rare for two or more customers to have the same name.

Explanation Prepared By: 'Afif Azhan Mohd Ismail
Code Prepared By: Arif Amiruddin bin Sadiran

Module 5 - Data Deletion

```
------------------------------------------------------------
                    DATA DELETION
------------------------------------------------------------

Current List :

Customers List
--------------
No  Name              Age   Identification Card   Date  Month     Sport Type     Check In  Check Out
 1  Harith Hamizan    20    001015056901          16    May       Tennis         800       1100
 2  Ainul Daniesya    31    890130148814          14    July      Ping Pong      900       1100
 3  Pavithra Rajan    37    831118035172          8     January   Bowling        1600      1900
 4  Looi Jay          42    780527023048          9     February  Futsal         2000      2300
 5  Uzair Hakimi      25    950304011217          21    December  Badminton      2000      2300
 6  Riveena Andria    32    880715100366          18    November  Swimming       1400      1700
 7  Wing Xing         18    020217081158          29    September Bowling        1100      1400
 8  Nurul Ain         24    960616148842          15    Mac       Netball        1500      1800
 9  Saiful Badri      19    010409018733          1     April     Badminton      800       1100
 10 Farhan Amir       28    921228031859          3     July      Futsal         2100      2300

Select which customer record you want to delete ('0' to return)
Option :
```

The screen above shows the data deletion function menu, where the user is able to choose to delete a customer details from the list. The user only needs to enter an integer that matches the customer number on the left of the list, then the customer will be deleted.

```
------------------------------------------------------------
                    DATA DELETION
------------------------------------------------------------

Customers List
--------------
No  Name              Age   Identification Card   Date  Month     Sport Type     Check In  Check Out
 1  Harith Hamizan    20    001015056901          16    May       Tennis         800       1100
 2  Ainul Daniesya    31    890130148814          14    July      Ping Pong      900       1100
 3  Pavithra Rajan    37    831118035172          8     January   Bowling        1600      1900
 4  Looi Jay          42    780527023048          9     February  Futsal         2000      2300
 5  Uzair Hakimi      25    950304011217          21    December  Badminton      2000      2300
 6  Riveena Andria    32    880715100366          18    November  Swimming       1400      1700
 7  Wing Xing         18    020217081158          29    September Bowling        1100      1400
 8  Nurul Ain         24    960616148842          15    Mac       Netball        1500      1800
 9  Saiful Badri      19    010409018733          1     April     Badminton      800       1100
 10 Farhan Amir       28    921228031859          3     July      Futsal         2100      2300

Select which customer record you want to delete ('0' to return)
Option : 11

There is no number '11' in the list. Please try again.
```

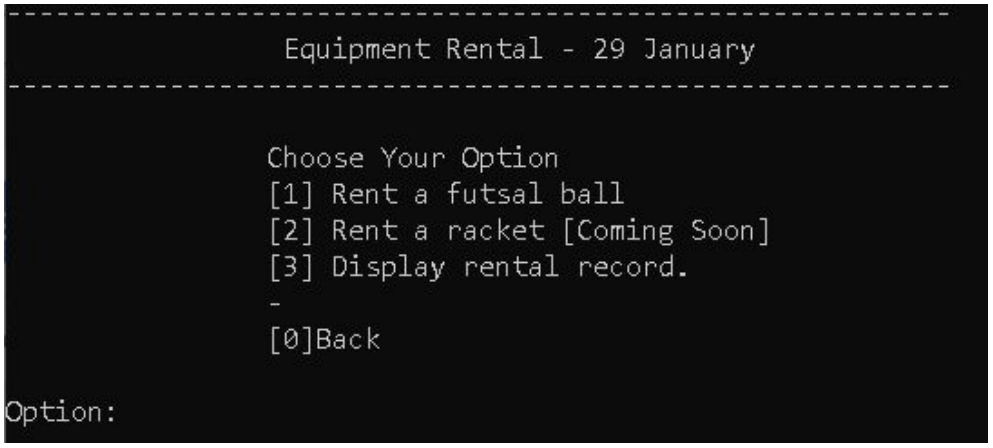
If the user entered a non-existent integer, the program will prompt an error message and the screen will be displayed again. To return to the main menu, the user needed to enter 0 in the option input.

```
Select which customer record you want to delete ('0' to return)
Option : 1

Delete the record for Harith Hamizan ? Option (Y/N) : Y

The list after deletion process :

Customers List
--------------
No  Name              Age   Identification Card   Date  Month     Sport Type     Check In  Check Out
 1  Ainul Daniesya    31    890130148814          14    July      Ping Pong      900       1100
 2  Pavithra Rajan    37    831118035172          8     January   Bowling        1600      1900
 3  Looi Jay          42    780527023048          9     February  Futsal         2000      2300
 4  Uzair Hakimi      25    950304011217          21    December  Badminton      2000      2300
 5  Riveena Andria    32    880715100366          18    November  Swimming       1400      1700
 6  Wing Xing         18    020217081158          29    September Bowling        1100      1400
 7  Nurul Ain         24    960616148842          15    Mac       Netball        1500      1800
 8  Saiful Badri      19    010409018733          1     April     Badminton      800       1100
 9  Farhan Amir       28    921228031859          3     July      Futsal         2100      2300
```

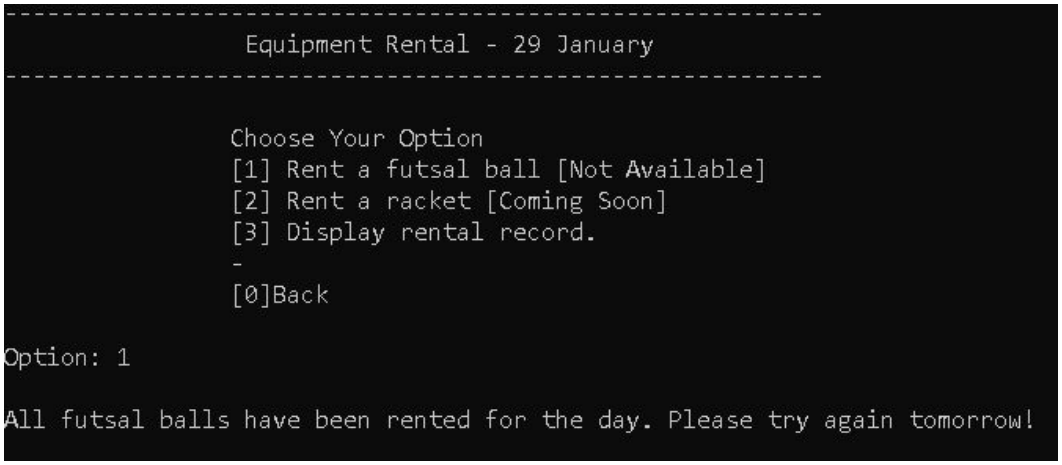
When the user enters a valid integer, the program will ask for the user's confirmation to either delete the details of the chosen customer or not. If he agreed, then the program will display the new list afterwards. Otherwise, the main menu of the deletion function will be displayed. After deletion, the deletion function main menu will be displayed again.

Explanation Prepared By: 'Afif Azhan Mohd Ismail
Code Prepared By: 'Afif Azhan Mohd Ismail

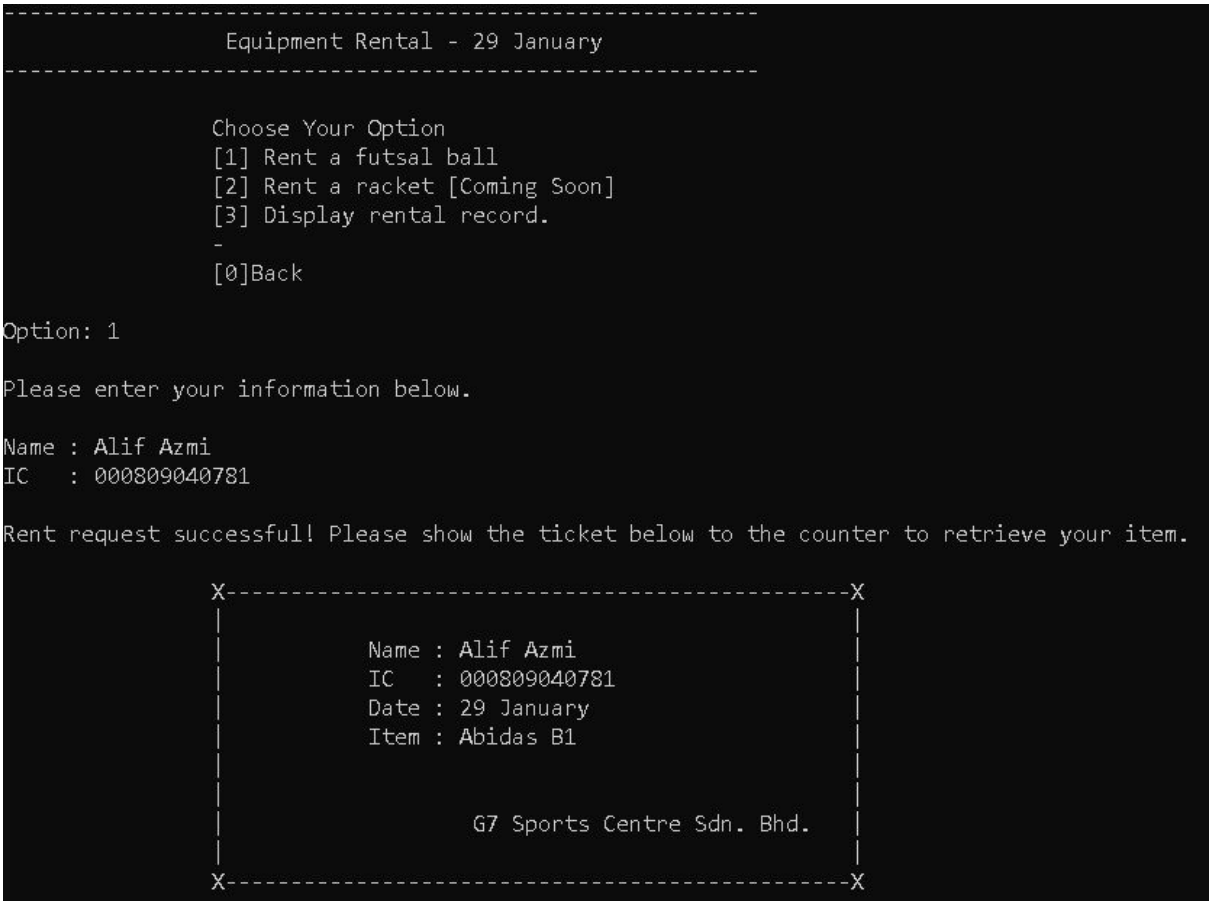
Module 6 - Daily Equipment Rental



For module 6, the figure above shows the main screen of the equipment rental function. The customers are only allowed to see the rent function while the admins can see the 3rd option. The title of the function shows the rent function for 29th January which is the date when the program is run. That means every day has a different rent function. To rent an equipment, users can choose either 1 or 2 depending on the availability of the equipment. If the user wants to see the rental record, user insert integer 3 in the option. To return to the main menu, the user needs to enter integer 0. At this moment, our sport centre does not provide racket rental service, so option 2 is still not available.



If all available equipment is already rented for the day, the screen will display “Not available” next to the option. If the user still chose the option, a message saying all equipment has been rented will show up.



When the user chose to rent an item, like in this example, the user chose to rent a futsal ball, the display will ask him to enter renter's name and IC number. The program will display a rent ticket for the customer, where they need to show to the counter to receive their items.

Explanation Prepared By: 'Afif Azhan Mohd Ismail

Code Prepared By: 'Afif Azhan Mohd Ismail

PART 4: DEVELOPMENT ACTIVITIES

Online Meeting Date	Members Participate in the meeting	Activity	Task for each member	Task Achieved (Yes/No)
20/1/2021 10pm	Arif 'Afif Iskandar	Choosing leader Brainstorming regarding project information and what to be done Trying to implement certain concept such as queue and stack in the program Determining additional functions that are related to program for our sports center to ease users' usage	**Arif** Initiate the meeting and suggest few concepts **'Afif** Relating few things from previous assignment from the project **Iskandar** Jot down discussions and produce report template via collaboration tools	Yes
25/1/2021 4.30pm	Arif 'Afif Iskandar	Distributing tasks regarding the project report Deeper discussion to implement queue concept in our program	**Arif** Explaining system program for module 3,4 and 5 regarding data sorting, searching and deletion **'Afif** Explaining system program for module 6 regarding daily requirement rent which apply queue concept **Iskandar** Explaining system program for module 1 and 5 regarding data display and data insertion	Yes

29/1/2021 1.00 pm	Arif 'Afif Iskandar	Finalizing project distribution by assigning certain things that are still not done yet such as flowcharts, conclusion. Begins to record the presentation video for the project. Divide topic on who is going to explain which part of the video. Deciding who will compile the videos and publish it on YouTube for easier submission via elearning.	**Arif** Recording video presentation for module 3,4 and 5 as well as system prototype. Volunteer to compile all video presentation from other team member **'Afif** Recording video presentation for module 6 as well as conclusion **Iskandar** Completing presentation slide by extracting information from main report Recording video presentations regarding module 1 and 2 explanation and also system analysis and design.	Yes
30/1/2021 3pm	Arif 'Afif Iskandar	Final touch up on project report documentation, presentation slides, source codes and video presentation.	**Arif** Finalizing video presentation to be uploaded to YouTube **'Afif** Finalizing project report **Iskandar** Finalizing slides	Yes

PART 5: APPENDIX
5. 1 Source Code

```
#include <iostream>
#include <fstream>
#include <string>
#include <cstring>
#include <iomanip>
#include <conio.h>
#include <ctime>
#include <fstream>
using namespace std;

string Month[12] =
{"January","February","March","April","May","June","July","August","September","Octobe
r","November","December"};
string Sports[7] = {"Futsal","Badminton","Netball","Bowling","Swimming","Ping
Pong","Tennis"};
const int size = 10;
time_t     rawtime;
struct tm* timeinfo;
ofstream out;
ifstream bout;

class Node
{
public:

   string name,sportType,month,ic,borrowedItem;
   int age,date,chin,chout,mon;
   Node* link; // pointer to next node
};

class EquipmentList
{
        private:
                int front, back,count;
                string items[size];

        public:
                EquipmentList()
                {
                        front = 0;
                        back  = size-1;
                        count = 0;
                }
```

```
        ~EquipmentList()
        {
                delete [] items;
        }

        bool isEmpty()
        {
                return(count == 0);
        }

        void insertEq(string newItem)
        {
                back = (back+1) % size;
                items[back] = newItem;
                ++count;
        }

        string getFront()
        {
                return items[front];
        }

        void removeEq()
        {
                front = (front+1) % size;
                --count;
        }

};

EquipmentList Ball;

EquipmentList rentedBall;

class List
{
        private:
        Node* head;
        Node* borrowHead ;

        public:
        List(void) { head = NULL;
                borrowHead = NULL;}
        ~List(void) { head = NULL;
                borrowHead = NULL;};
```

```
        Node* InsertNode(string, string, string, int, string, int ,int ,int);
        Node* InsertNode(string, string, string, int, string, int ,int ,int,int);
        Node* InsertBorrowNode(string, string, int, int, string);
        void deleteNode(int);
        void Find();
        void FindName(string);
        void FindIC(string);
        void FindMonth(string);
        void Sort();
        void SortName(int);
        void SortIC(int);
        void SortAge(int);
        void Delete();
        void Insert();
        void Rent();
        void menu();
        int displayBorrower();
        int DisplayList();

};

Node* List::InsertNode(string n, string m, string s, int a, string i, int d, int ci, int co)
{
        int currIndex = 0;
        Node* currNode = head;
        Node* prevNode = NULL;
        while ((currNode && n>currNode->name)&&(currNode &&
s>currNode->sportType)&&(currNode && m>currNode->month)&&(currNode &&
a>currNode->age)&&
        (currNode && i>currNode->ic)&&(currNode && d>currNode->date)&&(currNode
&& ci>currNode->chin)&&(currNode && co>currNode->chout))
                {
        prevNode = currNode;
        currNode = currNode->link;
        currIndex++;
                }

        Node* newNode = new Node;
        newNode->name= n;
        newNode->sportType= s;
        newNode->month= m;
        newNode->age= a;
        newNode->ic= i;
        newNode->date= d;
        newNode->chin= ci;
        newNode->chout= co;
```

```
        if(currIndex==0)
        {
   newNode->link=head;
   head= newNode;
        }

        else {
        newNode->link= prevNode->link;
        prevNode->link= newNode;
        }

        return newNode;
}


Node* List::InsertNode(string n, string m, string s, int a, string i, int d, int ci, int co,int index)
{
        int currIndex = 1;
        Node* currNode = head;
        while ((currNode && index > currIndex))
                {
        currNode = currNode->link;
        currIndex++;
                }

        if(index>0 && currNode == NULL)
        return NULL;

        Node* newNode = new Node;
        newNode->name= n;
        newNode->sportType= s;
        newNode->month= m;
        newNode->age= a;
        newNode->ic= i;
        newNode->date= d;
        newNode->chin= ci;
        newNode->chout= co;

        if(index==0)
        {
   newNode->link=head;
   head = newNode;
        }

        else {
```

```
        newNode->link = currNode->link;
        currNode->link = newNode;
        }

        return newNode;
}

Node* List::InsertBorrowNode(string n, string i, int m, int d, string b)
{

        int currIndex = 0;
        Node* currNode = borrowHead;
        Node* prevNode = NULL;
        while ((currNode && n>currNode->name)&&(currNode &&
i>currNode->ic)&&(currNode && d>currNode->date))
                {
        prevNode = currNode;
        currNode = currNode->link;
        currIndex++;
                }

        Node* newNode = new Node;
        newNode->name= n;
        newNode->mon= m;
        newNode->ic= i;
        newNode->date= d;

        newNode->borrowedItem = b;

        if(currIndex==0)
        {
   newNode->link=borrowHead;
   borrowHead= newNode;
        }

        else {
        newNode->link= prevNode->link;
        prevNode->link= newNode;
        }

        return newNode;
}

int List::DisplayList()
{
        int num = 1;
```

```
	Node* currNode = head;

	if(currNode)
	{
		cout << "\nCustomers List"
			<< "\n--------------";
		cout << left << endl << setw(4) << "No" << setw(20) << "Name" << setw(6)
<< "Age" << setw(22) << "Identification Card"
				<< setw(6) << "Date" << setw(10) << "Month" << setw(15) <<
"Sport Type" << setw(6) << "Check In"
				<< setw(6) << "Check Out" << endl;
	}

	else
	{
		cout << "There are no customer record in the list.";
		return 0;
	}

	while(currNode != NULL)
	{

		cout << left << " " << setw(3) << num << setw(20) << currNode->name <<
setw(6) << currNode->age << setw(22) << currNode->ic
			<< setw(6) << currNode->date << setw(10) << currNode->month <<
setw(15)<< currNode->sportType
			<< setw(6) << currNode->chin << setw(6) << currNode->chout <<
endl;
	currNode = currNode->link;
		num++;
   }
   return num;
}

int List::displayBorrower()
{
	int num = 1;
	Node* currNode = borrowHead;

	if(!currNode)
	{
	cout << "\nNo one has rented any equipment today yet.";
	return 0;
	}

	else
```

```
	{
		cout << left << endl << setw(4) << "No" << setw(20) << "Name" << setw(22)
<< "Identification Card"
			<< setw(6) << "Date" << setw(10) << "Month"  << setw(5) <<
"Rented Item"<< endl;
	}

	while(currNode != NULL)
	{

		cout << left << " " << setw(3) << num << setw(20) << currNode->name <<
setw(22) << currNode->ic
			<< setw(6) << currNode->date << setw(10) <<
Month[currNode->mon] << currNode->borrowedItem << endl;
	currNode = currNode->link;
		num++;
   }
   return num;
}

void List::FindName(string N)
{
	Node* currNode = head;
	while (currNode  && currNode ->name != N) {
	currNode  = currNode ->link;
	}

	if (currNode)
		{
			cout << "Below is the information of " << N << ":" << endl << endl;
			cout << "Name  : " << currNode ->name << endl
				<< "Age   : " << currNode ->age << endl
				<< "IC    : " << currNode ->ic  << endl
				<< "Sport : " << currNode ->sportType << endl
				<< "Date  : " << currNode ->date << " " <<  currNode
->month << endl
				<< "Checked in time  : " << currNode ->chin << endl
				<< "Checked out time : " << currNode ->chout << endl<<endl;


		}

	else
		cout << "There is no customer named " << N << " in the record." << endl <<
endl;
		getch();
```

```
		return;
}

void List::FindIC(string I)
{
	Node* currNode = head;
	while (currNode && currNode->ic != I) {
	currNode = currNode->link;
	}

	if (currNode)
		{
			cout << "\nBelow is the information about the customer with IC " << I
<< " :" << endl <<endl ;
			cout << "Name  : " << currNode->name << endl
				<< "Age   : " << currNode->age << endl
				<< "IC    : " << currNode->ic  << endl
				<< "Sport : " << currNode->sportType << endl
				<< "Date  : " << currNode->date << " " <<  currNode->month
<< endl
				<< "Checked in time  : " << currNode->chin << endl
				<< "Checked out time : " << currNode->chout << endl<<endl;

		}

	else
		cout << "There is no customer with IC " << I << " in the record.\n";
		getch();
		return;
}

void List::FindMonth(string M)
{
	Node* currNode = head;
	int num = 0;
	while (currNode )
	{

	if (currNode->month == M)
		{
			num++;
			if (num == 1)
			{
				cout << "\nBelow is the customer(s) that used the sport centre
in " << M << " :" << endl << endl;
```

```
                                cout << left << setw(4) << "No" << setw(20) << "Name" <<
setw(6) << "Age" << setw(22) << "Identification Card"
                                << setw(6) << "Date" << setw(10) << "Month" << setw(15) <<
"Sport Type" << setw(6) << "Check In"
                                << setw(6) << "Check Out" << endl;
                        }

                        cout << left << setw(4) << num << setw(20) << currNode->name <<
setw(6) << currNode->age << setw(22) << currNode->ic
                         << setw(6) << currNode->date << setw(10) << currNode->month <<
setw(15)<< currNode->sportType
                         << setw(6) << currNode->chin << setw(6) << currNode->chout <<
endl;
                }
        currNode = currNode->link;
        }
                if(num == 0)
                cout << "There is no customer enter the sport centre in " << M << ".";
                cout << endl << endl;
                getch();
                return ;
}

void List::Find()
{
        int ch,m;
        string tempIC,tempName,tempMonth;
   do
        {
                system("cls");
                cout << "------------------------------------------------------------\n";
                cout << "                    DATA SEARCHING\n";
                cout << "------------------------------------------------------------\n";

                cout << "\n\t\tChoose Your Option\n"
             << "\t\t[1] Search by Name"
                        << "\n\t\t[2] Search by Identification Card"
                        << "\n\t\t[3] Search by Month"
                        <<"\n\t\t-\n\t\t[0]Back\n\n";
            cout << "Option: ";
            cin  >> ch;
            switch(ch)
            {
               case 1 :
                  cin.ignore();
                                cout << "\nEnter Name: ";
```

```
                    getline(cin,tempName,'\n');
                    cout << endl;
   FindName(tempName);
   break;

case 2:

                    cout << "\nEnter Identification Card: ";
                    cin  >> tempIC;
                    FindIC(tempIC);
                    break;

case 3:
   cout << "\n\t\t\nChoose which month";
   cout << "\n\t\t[1] January"
            << "\n\t\t[2] February"
            << "\n\t\t[3] March"
            << "\n\t\t[4] April"
            << "\n\t\t[5] May"
            << "\n\t\t[6] June"
            << "\n\t\t[7] July"
            << "\n\t\t[8] August"
            << "\n\t\t[9] September"
            << "\n\t\t[10] October"
            << "\n\t\t[11] November"
            << "\n\t\t[12] December"
            << "\n\t\t-"
            << "\n\t\t[0] Back";
                    cout << "\n\nOption: ";
                    cin >> m;

                    if(m == 0)
                    Find();

                    if(m < 1 || m > 12)
                    {
                    cout << "Invalid option ! Please try again." << endl;
   getch();
   Find();
            }
                    FindMonth(Month[m-1]);
   break;

            case 0 :
            return;
```

```
                default :
                    cout << "\nInvalid option ! Please try again." << endl;
                    getch();
            }
    }
        while(ch!=0);
}

void List::Delete()
{
                    int ch;
                    system("cls");
                    cout << "-------------------------------------------------------\n";
                    cout << "                    DATA DELETION\n";
                    cout << "-------------------------------------------------------\n";
                    cout << "\nCurrent List :\n";
                    DisplayList();
                    cout << "\nSelect which customer record you want to delete ('0' to
return)\nOption : ";
                    cin >> ch;
                    if(ch == 0)
                    menu();
                    deleteNode(ch);
                    cout << "\nThe list after deletion process :\n";
                    DisplayList();
                    cout << endl;
                    cout << "Returning to main menu.";
                    _getch();
              menu();
}

void List::deleteNode(int a)
{
        int currIndex = 1;
        char choice;
        Node* currNode = head;
        Node* prevNode = NULL;

        while (currNode && currIndex != a)
        {
        prevNode = currNode;
        currNode = currNode->link;
        currIndex++;
        }

        if(!currNode)
```

```
	{
		cout << "\nThere is no number '" << a << "' in the list. Returning to main
menu...\n\n";
		_getch();
		menu();
	}

	cout << "\nDelete the record for " << currNode->name << " ? Option (Y/N) : ";
	cin >> choice;

	if(choice == 'N')
	Delete();

	if (currNode) {
		if (prevNode)
			{
			prevNode->link = currNode->link;
			delete currNode;
			}

	else {
		head = currNode->link;
		delete currNode;
		}
		return ;
		}

	cout << "There is no number '" << a <<"' in the list.";
	return;
}

void List::Insert()
{
	string Iname,Imonth,Iic;
   int Iage,Idate,Ichin,Ichout,ch,Inum,IsportType;
	system("cls");
		cout << "------------------------------------------------------------\n";
		cout << "                    DATA INSERTION\n";
		cout << "------------------------------------------------------------\n";
		Inum = DisplayList();

		cout << "\n\n\t\tChoose Your Option\n"
		<< "\t\t[1] Insert data in the beginning of the list"
			<< "\n\t\t[2] Insert data in the middle of the list"
			<< "\n\t\t[3] Insert data in the end of the list"
			<<"\n\t\t-\n\t\t[0]Back\n\n";
```

```
cout << "Option: ";
cin  >> ch;

if(ch<0 || ch>3)
{
cout << "\nInvalid option! Please try again.";
getch();
Insert();
	}

	if(ch==0)
	menu();

cout << "\nPlease fill the information for the new data :\n";
cin.ignore();
cout << "\nName : ";
getline(cin,Iname,'\n');
cout << "Age  : ";
cin >> Iage;
cout << "IC   : ";
cin >> Iic;
cout << "Sport Type  : [1] Futsal\n"
	<< "              [2] Badminton\n"
	<< "              [3] Netball\n"
	<< "              [4] Bowling\n"
	<< "              [5] Swimming\n"
	<< "              [6] Ping Pong\n"
	<< "              [7] Tennis\n\n"
	<< "              Choice: ";
cin >> IsportType;
cout << "\nDate of Check In and Check Out (i.e.: 7 January) : ";
cin >> Idate;
cin >> Imonth;
cout << "Check In time (24H)  : ";
cin >> Ichin;
cout << "Check Out time (24H) : ";
cin >> Ichout;

		if(ch == 1)
		{
	InsertNode(Iname,Imonth,Sports[IsportType-1],Iage,Iic,Idate,Ichin,Ichout);
	}

   if(ch == 2)
		{
```

```
InsertNode(Iname,Imonth,Sports[IsportType-1],Iage,Iic,Idate,Ichin,Ichout,Inum/2-1);
                }

            if(ch == 3)
                        {

InsertNode(Iname,Imonth,Sports[IsportType-1],Iage,Iic,Idate,Ichin,Ichout,Inum-1);
                }

          cout << "\nThe list after insertion process.\n";
                DisplayList();

                cout << "\nReturning to main menu.";
                _getch();
                return;
}

void List::SortName(int choice)
{
if(choice==1)
{
   Node* dummy = new Node;
   Node* currNode = head;

   while (currNode != NULL)
   {
      Node* temp = currNode->link;
      Node* prevNode = dummy;
      Node* lk = dummy->link;

      while (lk != NULL)
      {

         if (lk->name > currNode->name)
         {

            break;
         }

         prevNode = lk;
         lk = lk->link;
      }
      currNode->link = lk;
      prevNode->link = currNode;
      currNode = temp;
```

```
    }
    head = dummy->link;
}
else if(choice==2)
{
    Node* dummy = new Node;
    Node* currNode = head;

    while (currNode != NULL)
    {
        Node* temp = currNode->link;
        Node* prevNode = dummy;
        Node* lk = dummy->link;

        while (lk != NULL)
        {
            if (lk->name < currNode->name)
            {
                break;
            }

            prevNode = lk;
            lk = lk->link;
        }
        currNode->link = lk;
        prevNode->link = currNode;
        currNode = temp;
    }
    head = dummy->link;
}
cout << "\nSorting process successful. Please refer to the new list below. \n";
DisplayList();
getch();
Sort();
}

void List::SortIC(int choice)
{
if(choice==1)
{
    Node* dummy = new Node;
    Node* currNode = head;

    while (currNode != NULL)
    {
        Node* temp = currNode->link;
```

```
        Node* prevNode = dummy;
        Node* lk = dummy->link;

        while (lk != NULL)
        {
            if (lk->ic > currNode->ic)
            {
                break;
            }

            prevNode = lk;
            lk = lk->link;
        }
        currNode->link = lk;
        prevNode->link = currNode;
        currNode = temp;
    }
    head = dummy->link;
}
else if(choice==2)
{
    Node* dummy = new Node;
    Node* currNode = head;

    while (currNode != NULL)
    {
        Node* temp = currNode->link;
        Node* prevNode = dummy;
        Node* lk = dummy->link;

        while (lk != NULL)
        {
            if (lk->ic < currNode->ic)
            {
                break;
            }

            prevNode = lk;
            lk = lk->link;
        }
        currNode->link = lk;
        prevNode->link = currNode;
        currNode = temp;
    }
    head = dummy->link;
}
```

```
cout << "\nSorting process successful. Please refer to the new list below. \n";
DisplayList();
_getch();
Sort();
}

void List::SortAge(int choice)
{
if(choice==1)
{
   Node* dummy = new Node;
   Node* currNode = head;

   while (currNode != NULL)
   {
      Node* temp = currNode->link;
      Node* prevNode = dummy;
      Node* lk = dummy->link;

      while (lk != NULL)
      {
         if (lk->age > currNode->age)
         {
            break;
         }

         prevNode = lk;
         lk = lk->link;
      }
      currNode->link = lk;
      prevNode->link = currNode;
      currNode = temp;
   }
   head = dummy->link;
}
else if(choice==2)
{
   Node* dummy = new Node;
   Node* currNode = head;

   while (currNode != NULL)
   {
      Node* temp = currNode->link;
      Node* prevNode = dummy;
      Node* lk = dummy->link;
```

```
        while (lk != NULL)
        {
            if (lk->age < currNode->age)
            {
                break;
            }

            prevNode = lk;
            lk = lk->link;
        }
        currNode->link = lk;
        prevNode->link = currNode;
        currNode = temp;
    }
    head = dummy->link;
}
cout << "\nSorting process successful. Please refer to the new list below. \n";
DisplayList();
_getch();
Sort();
}

void List::Rent()
{
        int ch,ch1;
        string bName,bIC,bDay,bMonth;
        system("cls");
        cout << "------------------------------------------------------------\n";
        cout << "                Equipment Rental - " << timeinfo->tm_mday << " " <<
Month[timeinfo->tm_mon];
        cout << "\n------------------------------------------------------------\n";
        cout << "\n\t\tChoose Your Option\n"
            << "\t\t[1] Rent a futsal ball";

        if(Ball.isEmpty())
        cout << " [Not Available]";

        cout << "\n\t\t[2] Rent a racket [Coming Soon]";
        cout << "\n\t\t[3] Display rental record.";
        cout << "\n\t\t-\n\t\t[0]Back\n\n"
            << "Option: ";
        cin  >> ch;

        if(ch==0)
        menu();
```

```
        else if(ch==1)
                {
                        if(Ball.isEmpty())
                        {
                        cout << "\nAll futsal balls have been rented for the day. Please try
again tomorrow!";
                        getch();
                        }

                        else
                        {
                        cin.ignore();
                        cout << "\nPlease enter your information below."
                                << "\n\nName : ";
                        getline(cin,bName,'\n');

                cout << "IC   : ";
                cin >> bIC;


InsertBorrowNode(bName,bIC,timeinfo->tm_mon,timeinfo->tm_mday,Ball.getFront());
                out.open("RentedItem.txt",ios::app);
                out << bName << endl << bIC << endl << Ball.getFront() << endl <<
timeinfo->tm_mday << endl << timeinfo->tm_mon+1 << endl << endl;
                out.close();
                rentedBall.insertEq(Ball.getFront());

                cout << "\nRent request successful! Please show the ticket below to the
counter to retrieve your item.\n";
                cout << left <<"\n\t\tX-----------------------------------------------X"
                    << "\n\t\t|                                               |"
                    << "\n\t\t|\t   Name : " << setw(30) << bName << "|"
                    << "\n\t\t|\t   IC   : " << setw(30) << bIC   << "|"
                    << "\n\t\t|\t   Date : " << setw(2) << timeinfo->tm_mday << " " <<
setw(27) << Month[timeinfo->tm_mon] << "|"
                    << "\n\t\t|\t   Item : " << setw(30) << Ball.getFront() << "|"
                    << "\n\t\t|                                               |"
                    << "\n\t\t|                                               |"
                    << "\n\t\t|                G7 Sports Centre Sdn. Bhd.   |"
                    << "\n\t\t|                                               |"
                    << "\n\t\tX-----------------------------------------------X";

                Ball.removeEq();
                }
                }
```

```
	else if(ch==3)
	{
		displayBorrower();
		cout << endl;
	}

	else
	cout << "\nInvalid option. Please try again.";
	_getch();
	Rent();
}

void List::Sort()
{
	int ch,ch1;
	system("cls");
	cout << "-------------------------------------------------------\n";
	cout << "                  DATA SORTING\n";
	cout << "-------------------------------------------------------\n";
	cout << "\n\t\tChoose Your Option\n"
	    << "\t\t[1] Sort by Name"
		<< "\n\t\t[2] Sort by Identification Card"
		<< "\n\t\t[3] Sort by Age"
		<< "\n\t\t-\n\t\t[0]Back\n\n"
	    << "Option: ";

	cin  >> ch;

	if(ch==0)
	menu();

	else if(ch==1)
		{
			cout << "\n\t\tChoose Your Option\n"
		 << "\t\t[1]Ascending Order\n\t\t[2]Descending Order\n\t\t-\n\t\t[0]Back\n\n";
		cout << "Option: ";
		cin  >> ch1;

		if(ch1==0)
		Sort();

		else if(ch1 == 1 || ch1 ==2)
		SortName(ch1);

		else
		cout << "\nInvalid option ! Returning to sorting menu.";
```

```
                _getch();
                }

        else if(ch==2)
                {
                        cout << "\n\t\tChoose Your Option\n"
                 << "\t\t[1]Ascending Order\n\t\t[2]Descending Order\n\t\t-\n\t\t[0]Back\n\n";
                cout << "Option: ";
                cin  >> ch1;

                if(ch1==0)
                Sort();

                else if(ch1 == 1 || ch1 ==2)
                SortIC(ch1);

                else
                cout << "\nInvalid option ! Returning to sorting menu.";
                _getch();
                }

        else if(ch==3)
                {
                        cout << "\n\t\tChoose Your Option\n"
                 << "\t\t[1]Ascending Order\n\t\t[2]Descending Order\n\t\t-\n\t\t[0]Back\n\n";
                cout << "Option: ";
                cin  >> ch1;

                if(ch1==0)
                Sort();

                else if(ch1 == 1 || ch1 ==2)
                SortAge(ch1);

                else
                cout << "\nInvalid option ! Returning to sorting menu.";
                }

        else
        cout << "\nInvalid option. Please try again.";
        _getch();
        Sort();
}

void List::menu()
{
```

```
int ch;

        do
        {
        system("cls");
        cout << "------------------------------------------------------------\n";
        cout << "                SPORTS CENTRE CUSTOMERS RECORD\n";
        cout << "------------------------------------------------------------\n";
        cout << "\n\t\tChoose Your Option\n"
      << "\t\t[1] Data Display"
      << "\n\t\t[2] Data Insertion"
                  << "\n\t\t[3] Data Sorting"
                  << "\n\t\t[4] Data Searching"
                  << "\n\t\t[5] Data Deletion"
                  << "\n\t\t[6] Daily Equipment Rental"
                  << "\n\t\t-\n\t\t[0]Exit\n\n";
   cout << "Option: ";
                  cin  >> ch;
             switch(ch)
             {
                case 1 :
                   DisplayList();
                   cout << endl;
                   getch();
                   break;

                case 2:
                   Insert();
                   break;

                        case 3:
                                 Sort();
                                 break;

                case 4 :
                   Find();
                   break;

                case 5:
                   Delete();
                   break;

                case 6:
                   Rent();
                   break;
```

```
                    case 0:
                            cout << endl <<  "Thank youuuuuuuu :)";

                            out.open("AvailableBall.txt",ios::out);

                            while(!Ball.isEmpty())
                            {
                            out << Ball.getFront() << endl;
                            Ball.removeEq();
                            }
                            out.close();

                            out.open("RentedBall.txt",ios::app);

                            while(!rentedBall.isEmpty())
                            {
                            out << rentedBall.getFront() << endl;
                            rentedBall.removeEq();
                            }
                            out.close();

                exit(0);

              default:
                cout << "\nInvalid option! Please try again.";
                _getch();

                }

        }while(ch!=0);
        return;
}

int main()
{
List cust;
string inpName,inpIC,inpItem,usedBall;
int inpD, inpM;

string ballID[size] = {"Abidas B1","Abidas B2","Abidas B3","Abidas B4","Abidas
B5","Naik B6","Naik B7","Naik B8","Naik B9","Naik B10"};
cust.InsertNode("Farhan Amir","July","Futsal",28,"921228031859",3,2100,2300);
cust.InsertNode("Saiful Badri","April","Badminton",19,"010409018733",1,800,1100);
cust.InsertNode("Nurul Ain","Mac","Netball",24,"960616148842",15,1500,1800);
cust.InsertNode("Wing Xing","September","Bowling",18,"020217081158",29,1100,1400);
```

```
cust.InsertNode("Riveena
Andria","November","Swimming",32,"880715100366",18,1400,1700);
cust.InsertNode("Uzair
Hakimi","December","Badminton",25,"950304011217",21,2000,2300);
cust.InsertNode("Looi Jay","February","Futsal",42,"780527023048",9,2000,2300);
cust.InsertNode("Pavithra Rajan","January","Bowling",37,"831118035172",8,1600,1900);
cust.InsertNode("Ainul Daniesya","July","Ping Pong",31,"890130148814",14,900,1100);
cust.InsertNode("Harith Hamizan","May","Tennis",20,"001015056901",16,800,1100);

//for(int i=0;i<size;i++)
//Ball.insertEq(ballID[i]);


// Get Local Time
time( &rawtime );
timeinfo = localtime( &rawtime );

// Get balls that are available for rent

bout.open("AvailableBall.txt");

while(bout.eof())
{

        getline(bout,usedBall,'\n');
        if(usedBall == "")
        continue;
        Ball.insertEq(usedBall);
}
bout.close();

// Get balls that are rented recently
bout.open("RentedBall.txt",ios::in);
while(bout.good())
{
        getline(bout,usedBall,'\n');
        rentedBall.insertEq(usedBall);
}
bout.close();

// Get list of Past Equipment Borrower
bout.open("RentedItem.txt",ios::in);
while(bout.good())
{
        getline(bout,inpName,'\n');
        if(inpName == "")
```

```
        continue;
        getline(bout,inpIC,'\n');
        getline(bout,inpItem,'\n');
        bout >> inpD;
        bout >> inpM;
        cust.InsertBorrowNode(inpName,inpIC,inpM-1,inpD,inpItem);

}
bout.close();

// Check if day has changed, if yes, put yesterday's rented ball into available balls for rent
queue
        if(inpD != timeinfo->tm_mday)
        {

        while(!rentedBall.isEmpty())
        {
        Ball.insertEq(rentedBall.getFront());
        rentedBall.removeEq();
        }

        out.open("RentedBall.txt",ios::out);
        out.close();
        }

cust.menu();

return 0;
}
```

---- END OF DOCUMENTATION -----