SCHOOL OF COMPUTING

FACULTY OF ENGINEERING

UNIVERSITI TEKNOLOGI MALAYSIA

SEMESTER 1 2020/2021

DATA STRUCTURE & ALGORITHMS (SECJ2013)

SECTION 09

MINI PROJECT DOCUMENTATION
HOTEL BOOKING SYSTEM

GROUP MEMBER	MATRIC NO
CHIAM WOOI CHIN	A19EC0034
GOH JO EY	A19EC0047
NG JING ER	A19EC0115
ONG YIN REN	A19EC0204

LECTURER: DR MOHAMAD ASHARI HAJI ALIAS
DATE: 28th JANUARY 2021

PART 1: INTRODUCTION

1.1 Synopsis Project

Our C++ mini project is the Hotel Booking System. The project is for the admin of the hotel to manage the room and the booking details of customers and for the customer to book the hotel room.

The type of data structures used in this system are sorting, linked list and queue. There are several classes in this system which are customer, bill and hotel room class. The admin can add, delete and check the availability of the hotel room with the linked list data structures which add or delete the room to the list. The room types have single, double, premium and deluxe. The customer can book, cancel the room and check the availability of the hotel room by updating the linked list of rooms. They have to fill in their personal details in order to book the room. The system updates the information of customers booking by linked list and sorts it by using the sorting technique. The customer booking is added to the queue of pending booking. After the customer booked the room, the admin can check the customer booking and review the booking information. The admin needs to confirm the pending booking and provide the bill for the customer by using queue data structure. The queue is for the pending booking to add to the bill queue for confirmation of booking and delete the last pending booking if want to cancel.

1.2 Objective of The Project

- For admin to manage the details of hotel room, booking and customers
- To simplify the booking process of customer

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

2.1 System Requirements

2.1.1 Use case diagram

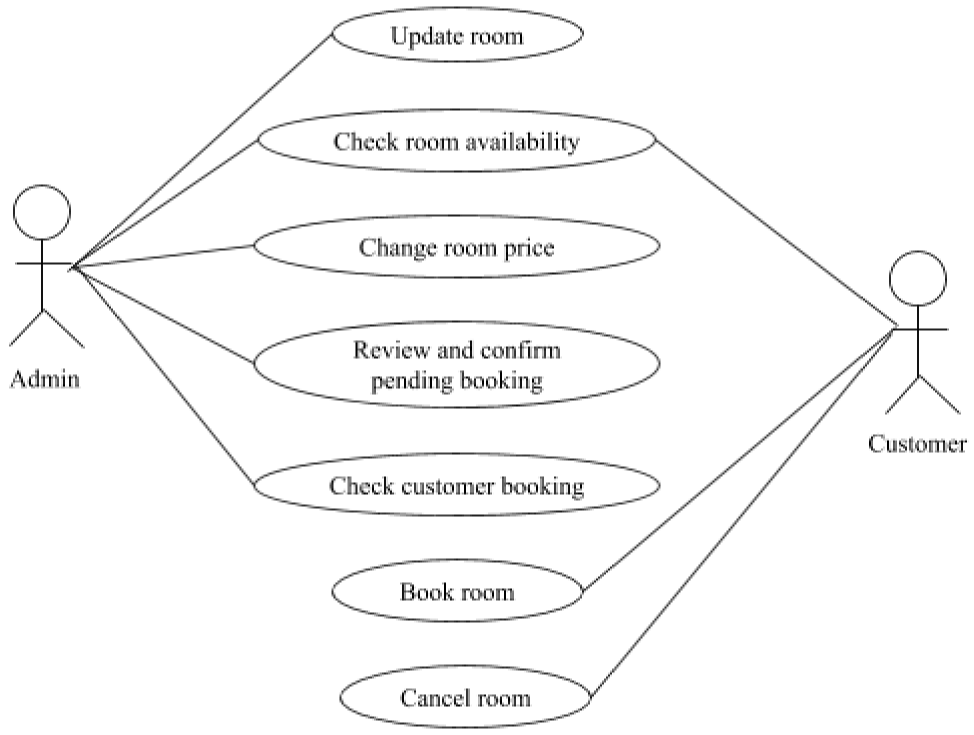


Figure 1: Use case diagram for Hotel Booking System

2.1.2 Use Cases Description for Hotel Booking System

The system users are admin and customer.

Actor	**Task**
Admin	The admin can add, delete and check the availability of the hotel room. The admin also can change the room price. After the customer booked the room, the admin can check the customer booking and review the booking

	information. The admin needs to confirm the pending booking and provide the bill for the customer.
Customer	The customer can book, cancel the room and check the availability of the hotel room. They have to fill in their personal details in order to book the room.

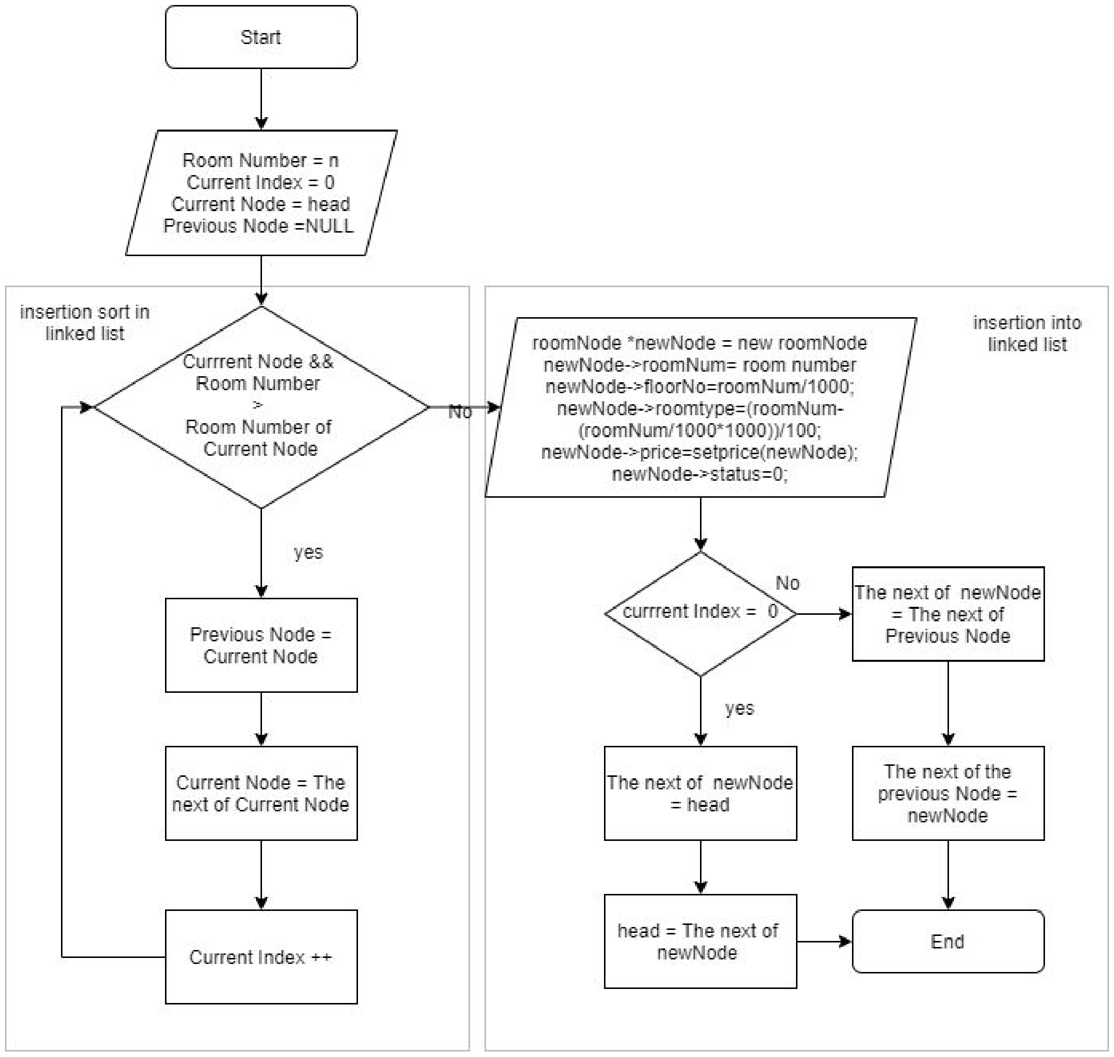
2.1.3 Detail Description for Each Use Cases

The system has 6 main use cases

Use Case	**Purpose**
Update room	Update information of room includes the updated room price by adding or deleting the room.
Check room availability	View the room information and availability of the room
Change room price	Change and update the price of each type of room
Check customer booking	View the detail list of the customer's booking
Review and confirm pending booking	Show all the details of the customers and the room before confirm the booking
Book room	Book the room and fill in their personal details by adding the booking to the booking list.
Cancel room	Make cancellation of their booking by deleting the booking from the booking list.

2.2 System Design

FlowChart 1: Add room (Admin - Update room)


Start
Room Number = n
Current Index = 0
Current Node = head
Previous Node =NULL
insertion sort in linked list
Currrent Node && Room Number > Room Number of Current Node
yes
No
Previous Node = Current Node
Current Node = The next of Current Node
Current Index ++
insertion into linked list
roomNode *newNode = new roomNode
newNode->roomNum= room number
newNode->floorNo=roomNum/1000;
newNode->roomtype=(roomNum-(roomNum/1000*1000))/100;
newNode->price=setprice(newNode);
newNode->status=0;
currrent Index = 0
No
yes
The next of newNode = The next of Previous Node
The next of the previous Node = newNode
The next of newNode = head
head = The next of newNode
End

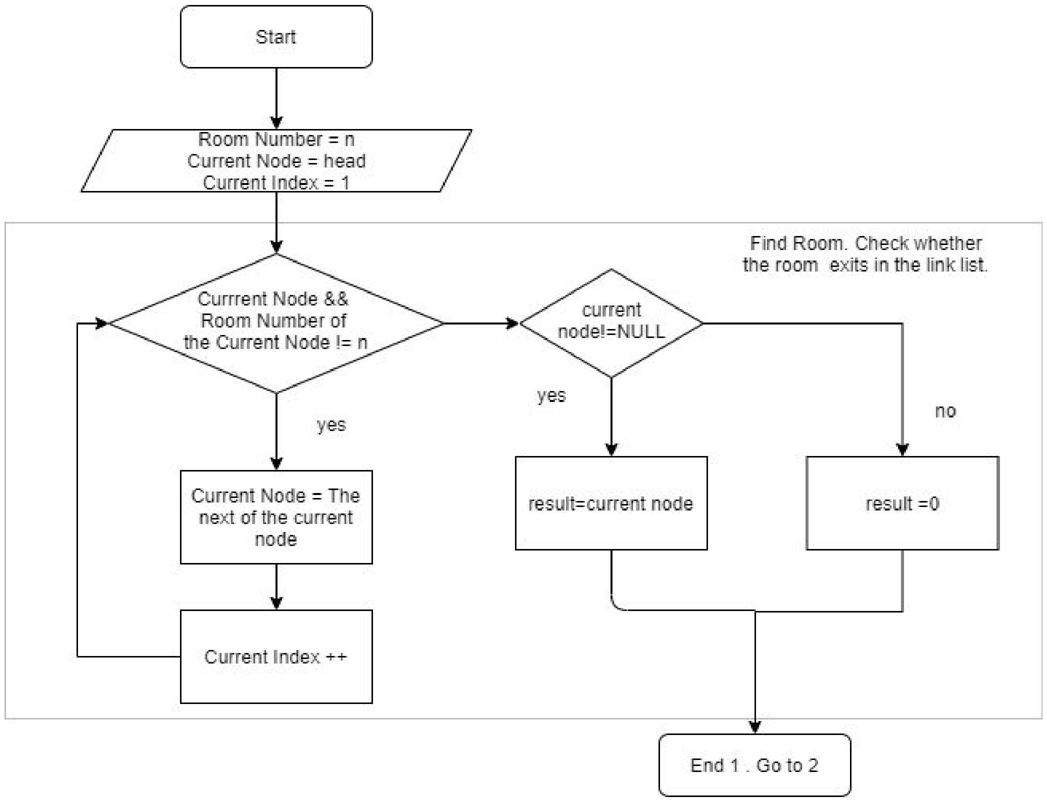
Explanation based on flowchart 1:

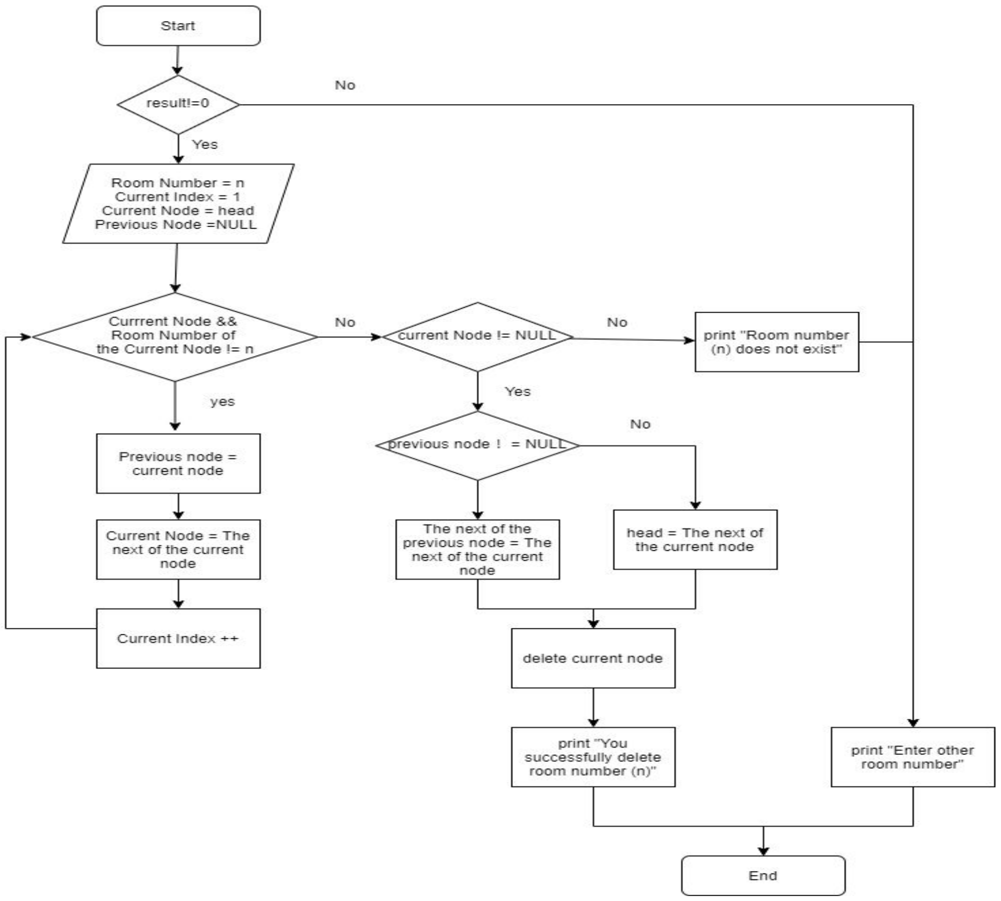
The data structure applied in this flowchart is insertion sort and linked list. The user is required to enter the room number that he/she wants to add to the system. It will first check whether the room number exists in the system or not. If the room number is unique, it will start the flow chart. The flowchart starts with initializing room number=n, current index = 0, points the current node to the head of the room linked list and the previous node set to NULL. The flowchart is divided into two-part, insertion sort and add a node into the room linked list. In the insertion sort part, it will find the correct position for the inserted node in ascending order of the room number. The flow chart starts to point current nodes to the head of the linked list. If the condition is true(the current node and the room number inserted by the user are greater than the room number of the current node), the loop will continue by changing the previousNode=currentNode, currentNode= the next of the current node and current index=current index+1. The loop will continue until the condition is false. When the condition is false, it will start the part to insert a node into the linked list. It starts with creating a new node. If the current index is equal to 0, it is an empty linked list, the next of the new node is pointing to the head(NULL) and the head is pointing to the new node. Otherwise, the next of the new node is pointing to the next of the previous node and the next of the previous node is pointing to the new node.

Prepared By: CHIAM WOOI CHIN & GOH JO EY

FlowChart 2: Delete room (Admin - Update room)

Data Structure: Delete node in linked


Start
Room Number = n
Current Node = head
Current Index = 1
Find Room. Check whether the room exits in the link list.
Currrent Node && Room Number of the Current Node != n
yes
Current Node = The next of the current node
Current Index ++
current node!=NULL
yes
no
result=current node
result =0
End 1 . Go to 2


Start
result!=0
No
Yes
Room Number = n
Current Index = 1
Current Node = head
Previous Node =NULL
Currrent Node && Room Number of the Current Node != n
No
yes
Previous node = current node
Current Node = The next of the current node
Current Index ++
current Node != NULL
No
Yes
print "Room number (n) does not exist"
previous node ! = NULL
No
The next of the previous node = The next of the current node
head = The next of the current node
delete current node
print "You successfully delete room number (n)"
print "Enter other room number"
End

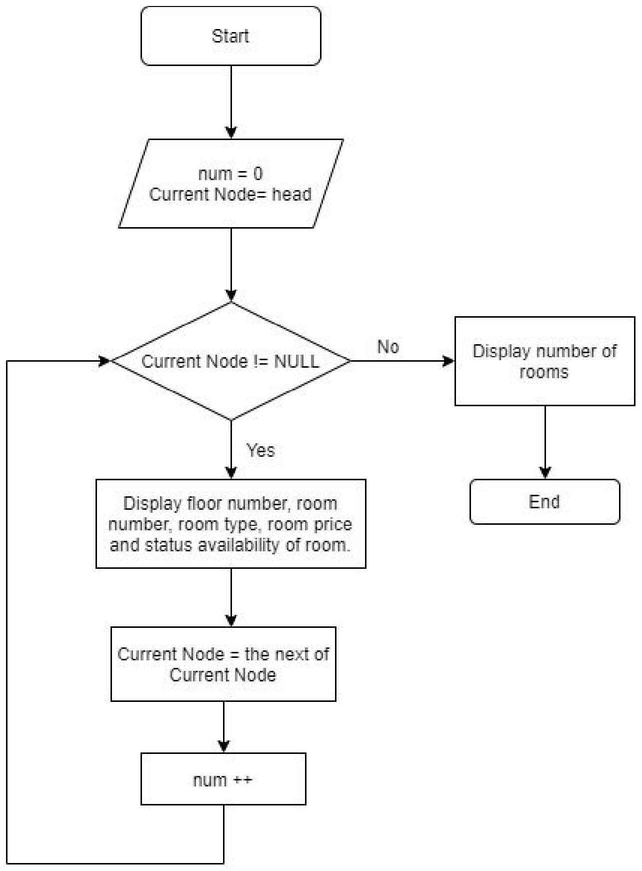
Explanation based on flowchart 2:

Delete a room from the system including two parts. First, it checks whether the room exits in the link list. If the first condition is true, it will continue to find the node in the linked list and delete it. For the first part, it will initialize room number as n, current node is pointing to the head of the linked list and current index equal to 1. If the current node and the room number of the current node is not equal to the room number entered by the user, it will continue to loop by pointing the current node to the next of the current node and increase the current index by 1. Once the loop is done, it will check whether the current node is Null or not. If it is null, it will return zero, otherwise, it will return the current node. For the second part, it will use the same theory as explained before to find the node that contains the room number entered by the user and delete the node. After delete the node, it will prompt a message of "You successfully delete the room number(n)". For the fail case(cannot find the corresponding node in the linked list), the system will prompt an error message.

Prepared By: CHIAM WOOI CHIN & GOH JO EY

FlowChart 3: Display room (Customer and Admin)

Data Structure:


Start
num = 0
Current Node= head
Current Node != NULL
No
Display number of rooms
End
Yes
Display floor number, room number, room type, room price and status availability of room.
Current Node = the next of Current Node
num ++

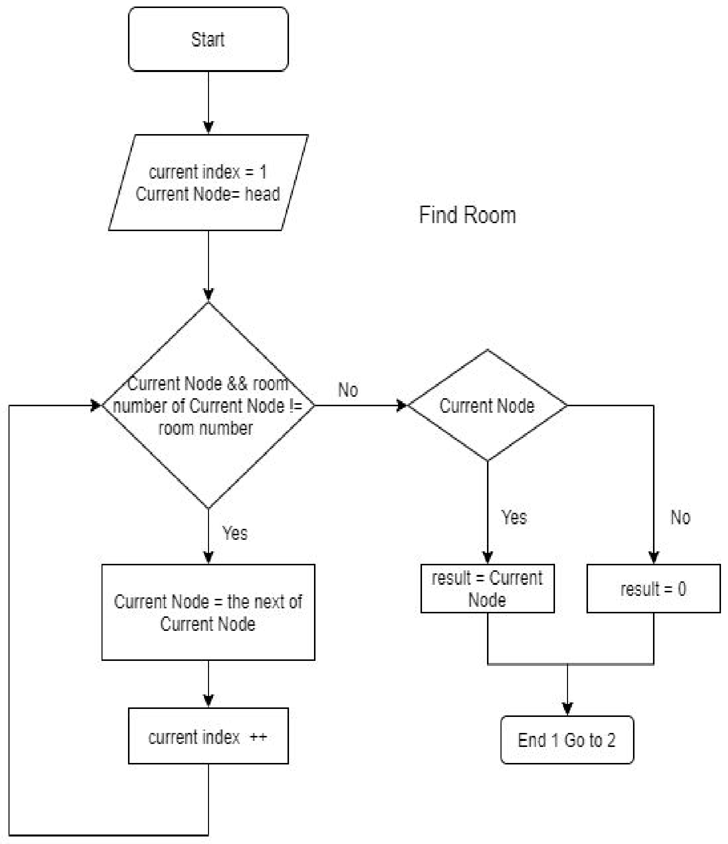
Explanation based on flowchart 3:

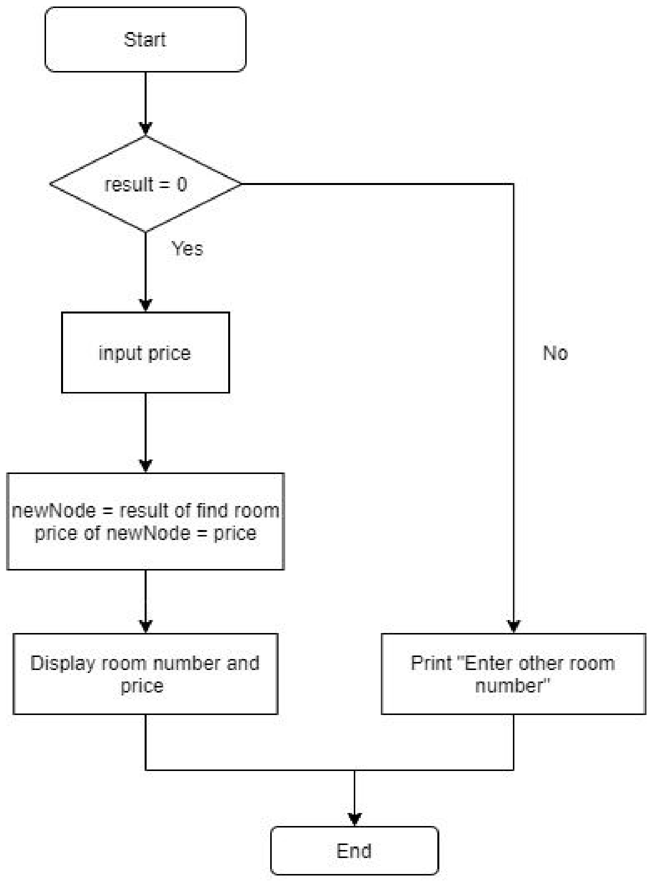
For the function to print the room in the hotel, it will apply the data structure of display in the linked list. The flowchart starts with initializing an integer to zero and pointing the current node to the head of the linked list. The loop will start and continue with the condition that the current node is not null. The loop will display the floor number, room number, room type, room price and status availability of the room. The loop is changing the current node by pointing the current node to the next node in the linked list and increasing the integer by 1. The loop is ended when the condition is false(current node is null).

Prepared By: CHIAM WOOI CHIN & GOH JO EY

FlowChart 4: Change price (Admin)

Data Structure: Delete node and find node in the linked list




Start
result = 0
Yes
No
input price
newNode = result of find room
price of newNode = price
Display room number and price
Print "Enter other room number"
End

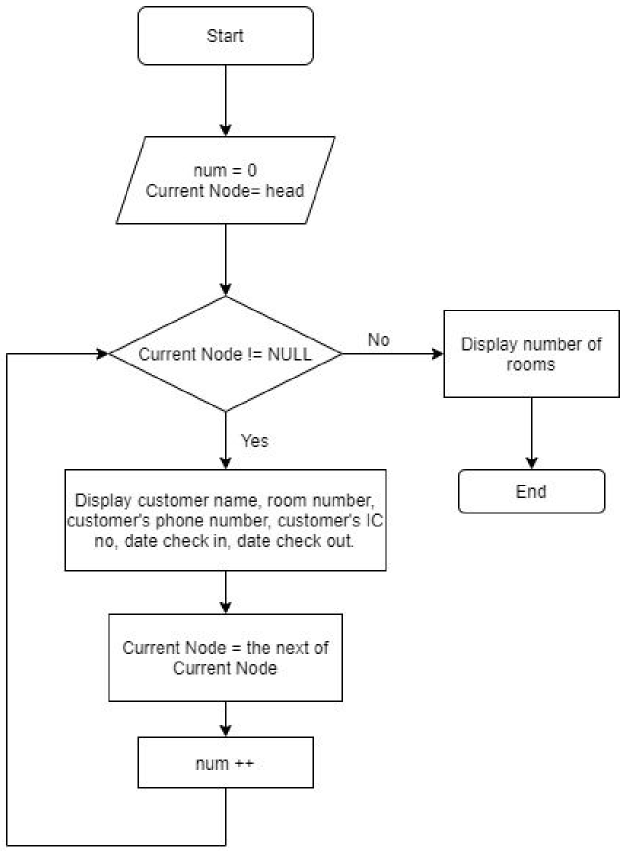
Explanation based on flowchart 4:

For the change in the price of the room by the admin, First, it checks whether the room exists in the linked list. If the first condition is true, it will continue to find the node in the linked list and change the price. For the first part, the current node is pointing to the head of the linked list and current index equal to 1. If the current node and the room number of the current node is not equal to the room number entered by the user, it will continue to loop by pointing the current node to the next of the current node and increase the current index by 1. Once the loop is done, it will check whether it is the current node or not. If it is, it will return zero, otherwise, it will return the current node. For the second part, if the room exists, it will let the admin input the price that wants to change. The newNode points to the result of find room which is the first part then the price of newNode points to the input price. After that, it will display the room number and the changed price. If the room does not exist, it will let the user enter another room number again.

Prepared By: CHIAM WOOI CHIN & GOH JO EY

FlowChart 5: Check customer booking list (Admin)

Data Structure:


Start
num = 0
Current Node= head
Current Node != NULL
No
Display number of rooms
End
Yes
Display customer name, room number, customer's phone number, customer's IC no, date check in, date check out.
Current Node = the next of Current Node
num ++

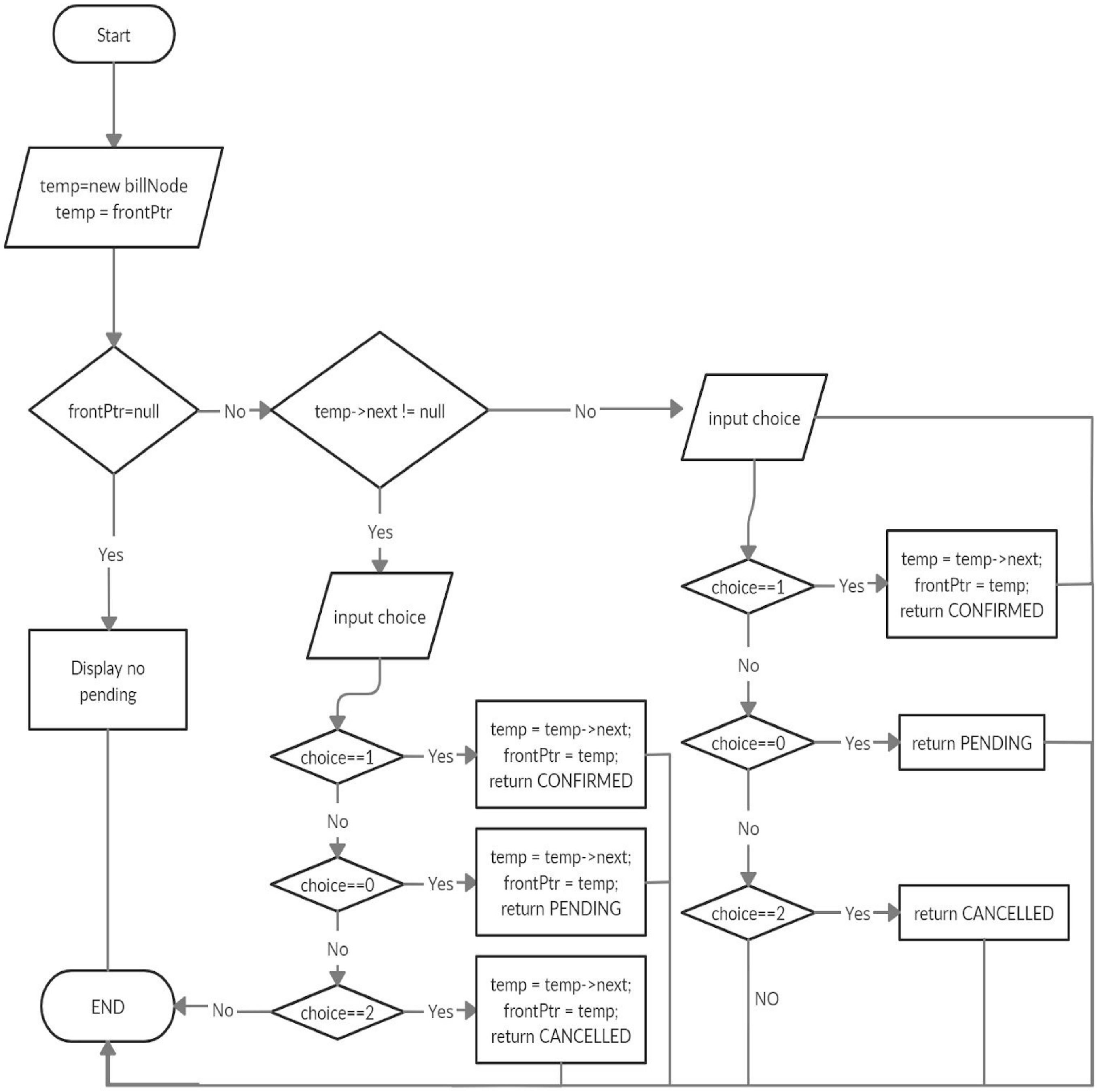
Explanation for flowchart 5:

Once the customer books a room from the system, it will create a customer linked list to store the information of the customers with their corresponding booked room. Admin have a function of checking the customer booking list in the system. If the customers cancel the room, the customers records and their booked room will be deleted. The flowchart starts to initialize num to zero and points the current node to the head. A loop will be started with a condition that the current node is not null. Inside the loop, it will display the information in the customer node, point the current node to the next node in the linked list, and increment num by 1. The flowchart is ended when the condition of the loop is false and displays the total number of rooms in the system.

Prepared By: NG JING ER & ONG YIN REN

FlowChart 6: Review and confirm pending booking (Admin)

Data Structure: Queue



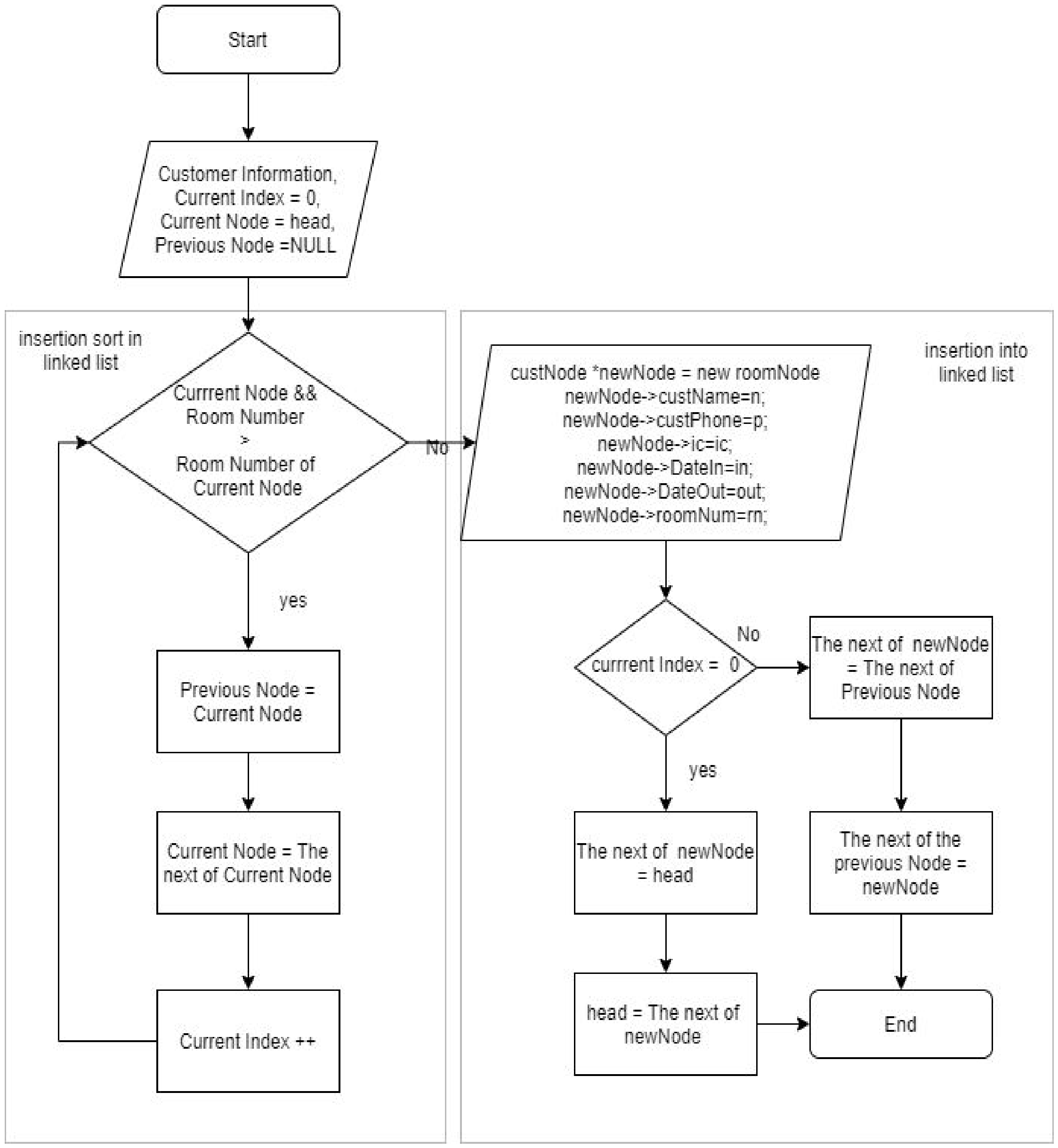
Explanation based on flowchart 6:

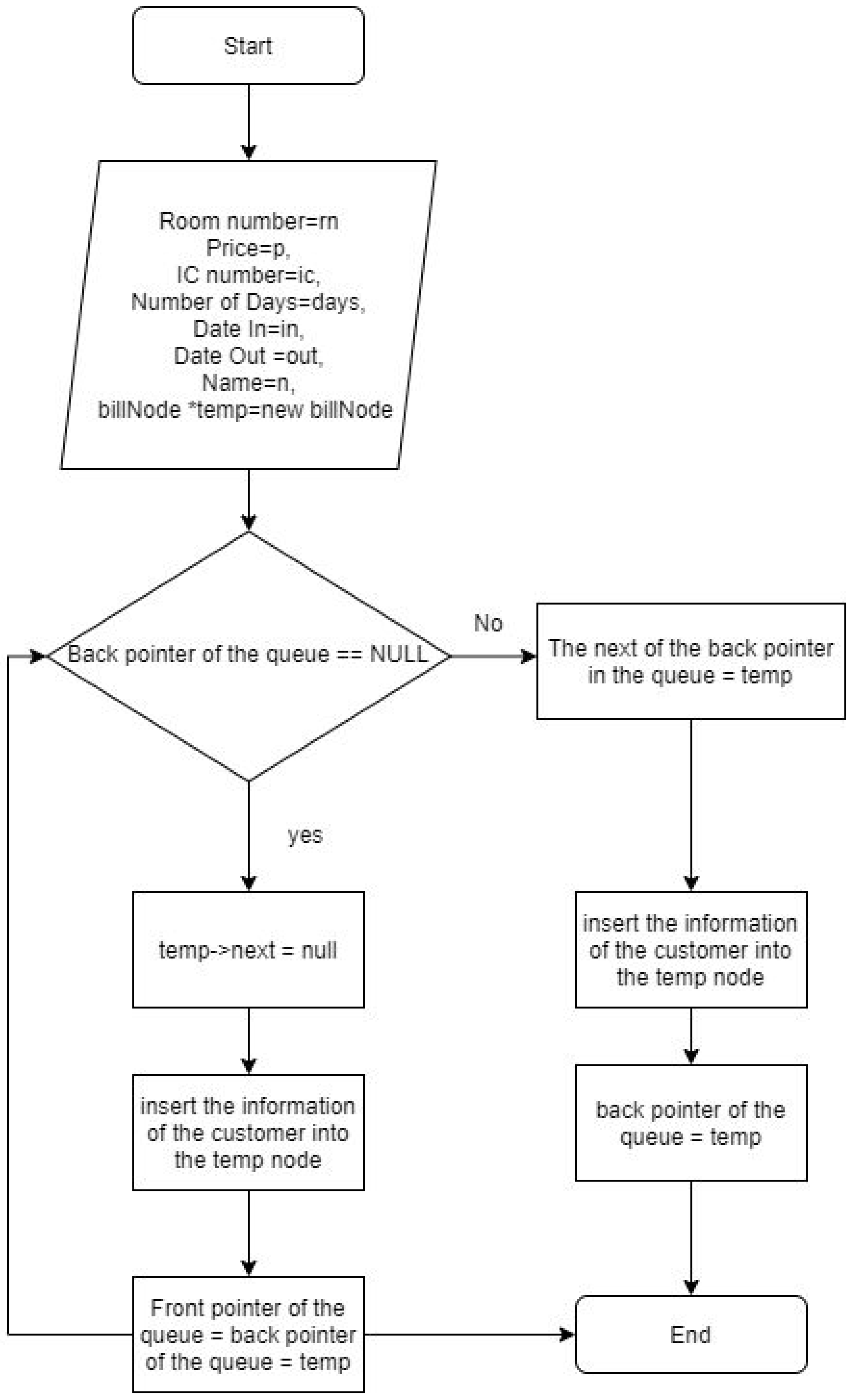
The review and confirm pending booking involve the implementation of a queue. The flowchart of this module is mainly about the function of deQueue() that under the class billQueue. The deQueue() function is used to remove and change the status of the rooms in the hotel. The new billNode is dynamically allocated and assigned to pointer temp, the, frontPtr is assigned into location pointed to by pointer temp.First, if the frontPtr is null, it will display to the user that there is no pending booking for review. Else if the next pointer in billNode is not null, the system will request the user to change the status of booking. To confirm the booking, the temp will be assigned to be the next pointer of billNode, and frontPtr will be assigned to be temp which may result in the pending booking to be removed from the queue. Or else if the case does not match with the above condition, the system will request the user to change the status of booking and only the status of the booking that is "CONFIRMED" will be removed from the queue.

Prepared By: NG JING ER & ONG YIN REN

Flowchart 7: Book Room

Data Structure: linked list and queue


Start
Customer Information, Current Index = 0, Current Node = head, Previous Node =NULL
insertion sort in linked list
Currrent Node && Room Number > Room Number of Current Node
No
yes
Previous Node = Current Node
Current Node = The next of Current Node
Current Index ++
insertion into linked list
custNode *newNode = new roomNode
newNode->custName=n;
newNode->custPhone=p;
newNode->ic=ic;
newNode->DateIn=in;
newNode->DateOut=out;
newNode->roomNum=rn;
currrent Index = 0
No
yes
The next of newNode = The next of Previous Node
The next of the previous Node = newNode
The next of newNode = head
head = The next of newNode
End


Start
Room number=rn
Price=p,
IC number=ic,
Number of Days=days,
Date In=in,
Date Out =out,
Name=n,
billNode *temp=new billNode
Back pointer of the queue == NULL
No
The next of the back pointer in the queue = temp
yes
temp->next = null
insert the information of the customer into the temp node
insert the information of the customer into the temp node
back pointer of the queue = temp
Front pointer of the queue = back pointer of the queue = temp
End

Explanation based on flowchart 7:

The book room implemented the linked list and queue.For the linked list implementation, first, it checks whether the room number is exits.For the first part, it will initialize the current index equal to 0, current node pointing to the head of linked list and previous node pointing to the NULL of linked list. If the condition of current node and the room number are bigger than the room number of the current node, it will continue looping by pointing previous node is equal to the node and the current node is equal to the next of the current node and last increases the current index by 1.For the second part, if the condition are false, the new node that reference to the custnode is equal to the new room node. It will initialize the new node pointer to custname as n, new node pointer to custphone as p, newnode pointer to ic are ic, newnode pointer to datein as in, newnode pointer to dateout as out an newnode pointer to roomnum as rn.

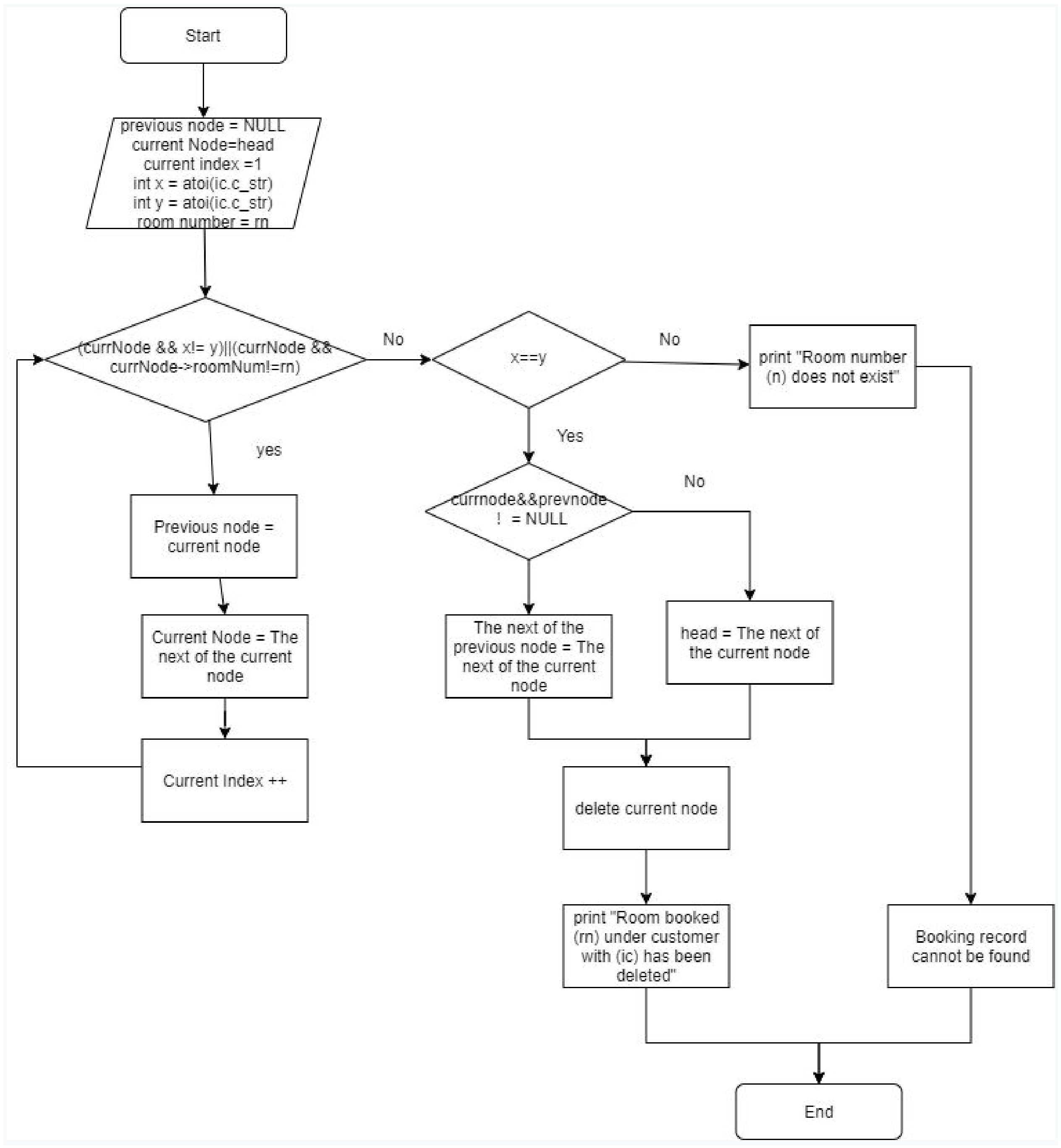
It will continue the loop by pointing that the current index is equal to 0. If this condition are true, it will continue looping by pointing the next of newnode equal to head and the head is equal to the next of newnode and end the booking process.However, it will continue looping by pointing the next of newnode is equal to the next of previous node if the condition is false.Then, it will continue looping by pointing the next of the previous node equal newnode and end the booking process.

For the implementation of queue, it initializes the room number as rn, price as p, ic number as ic, number of days as days, datein as in, dateout as out, name as n and the temp reference to billNode equal to new billNode.If the back pointer of the queue equal to NULL, it will continue looping by pointing the temp point to next equal to NULL.Then,user needs to insert the information of the customer into the temp node and front pointer of the queue equal to back pointer of the queue equal to temp.It will end the process or continue regarding the choice of user. However, if the back pointer of the queue is not equal to NULL, the next of the back pointer in the queue equal to temp.Then, user needs to insert the information of the customer into the temp node, the back pointer of queue equal to temp and last ended the flowchart.

Prepared By: NG JING ER & ONG YIN REN

Flowchart 8: Cancel Room

Data Structure: Delete node in linked list


Start
previous node = NULL
current Node=head
current index =1
int x = atoi(ic.c_str)
int y = atoi(ic.c_str)
room number = rn
(currNode && x!= y)||(currNode && currNode->roomNum!=rn)
No
yes
Previous node = current node
Current Node = The next of the current node
Current Index ++
x==y
No
Yes
print "Room number (n) does not exist"
currnode&&prevnode ! = NULL
No
The next of the previous node = The next of the current node
head = The next of the current node
delete current node
print "Room booked (rn) under customer with (ic) has been deleted"
Booking record cannot be found
End

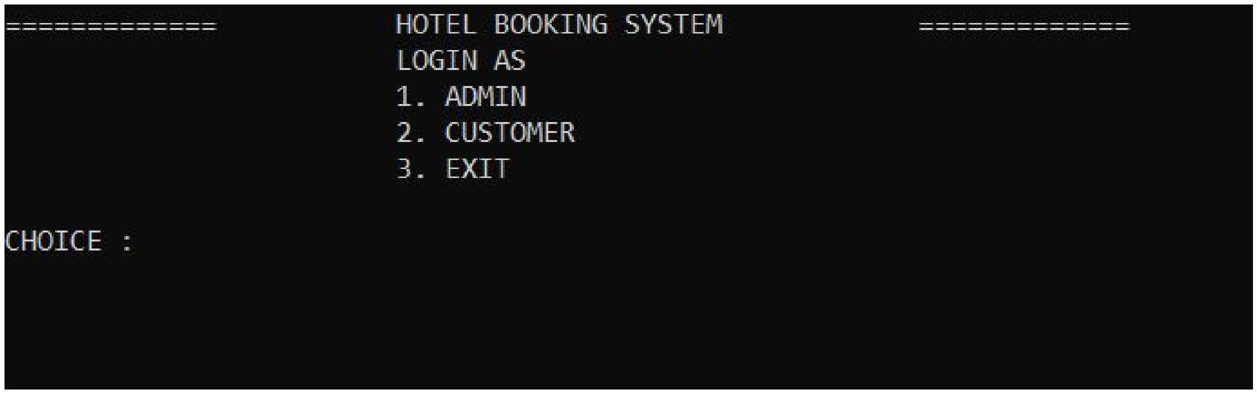
Explanation based on flowchart 8:

For the cancel booking room function, it will first start the find room function and check the room number entered by the user exits in the system and check whether the room is booked. It will only start the flowchart above(delete the customer node in the customer linked list) if the condition(the room exists and is booked) is true. The flowchart will initialize current index as 1 and string rn as the room number. It will point the previous node as NULL, and point the current node to the head of the linked list. It will start a condition to find the position of the finding node by comparing and checking the room number entered by the user. The loop will be continued by pointing previous node to the current node, pointing the current node to the next of the current node and increasing the current index by 1. After the loop is done it will check whether the ic entered by the user is the same with the ic in the found node. If the condition is true, it will start to point the current node to the next of current node(found node in middle or last of the linked list) or point the head to the next of the current node(found node in the first of the linked list) and delete the node. It is done with delete customer nodes in the customer linked list. Lastly, it will change the status of the room node to “available” if the delete customer node process is successful.

Prepared By: NG JING ER & ONG YIN REN

PART 3: SYSTEM PROTOTYPE

Below are some of the interfaces of our hotel booking system prototype. The users include admin and customer, therefore the interfaces below will be divided based on each user.

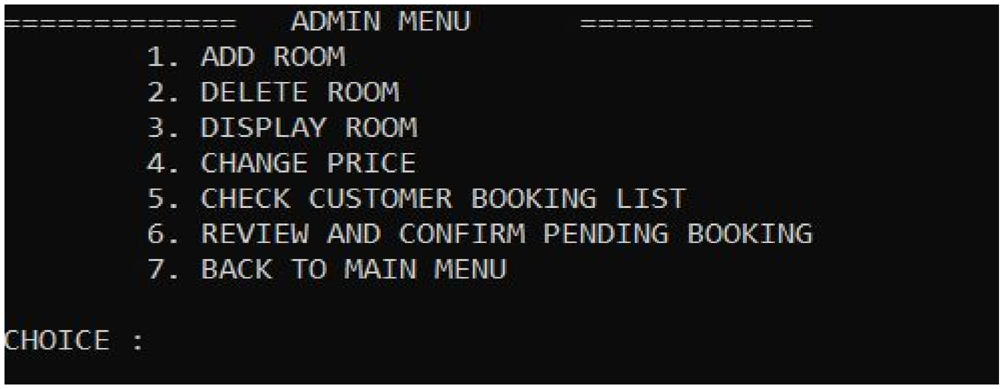


```
=============           HOTEL BOOKING SYSTEM           =============
                        LOGIN AS
                        1. ADMIN
                        2. CUSTOMER
                        3. EXIT

CHOICE :
```

Screen 1: Hotel Booking menu

Screen 1 : The user needs to enter the integer value in the range of 1 to 3.If the user enter 1 , user will login as an admin, and if user enter 2 will login as a customer and if user enter 3 it will exit the program. Otherwise, the program will prompt invalid choice and the screen will display again after the user presses any key.

Prepared By; ONG YIN REN

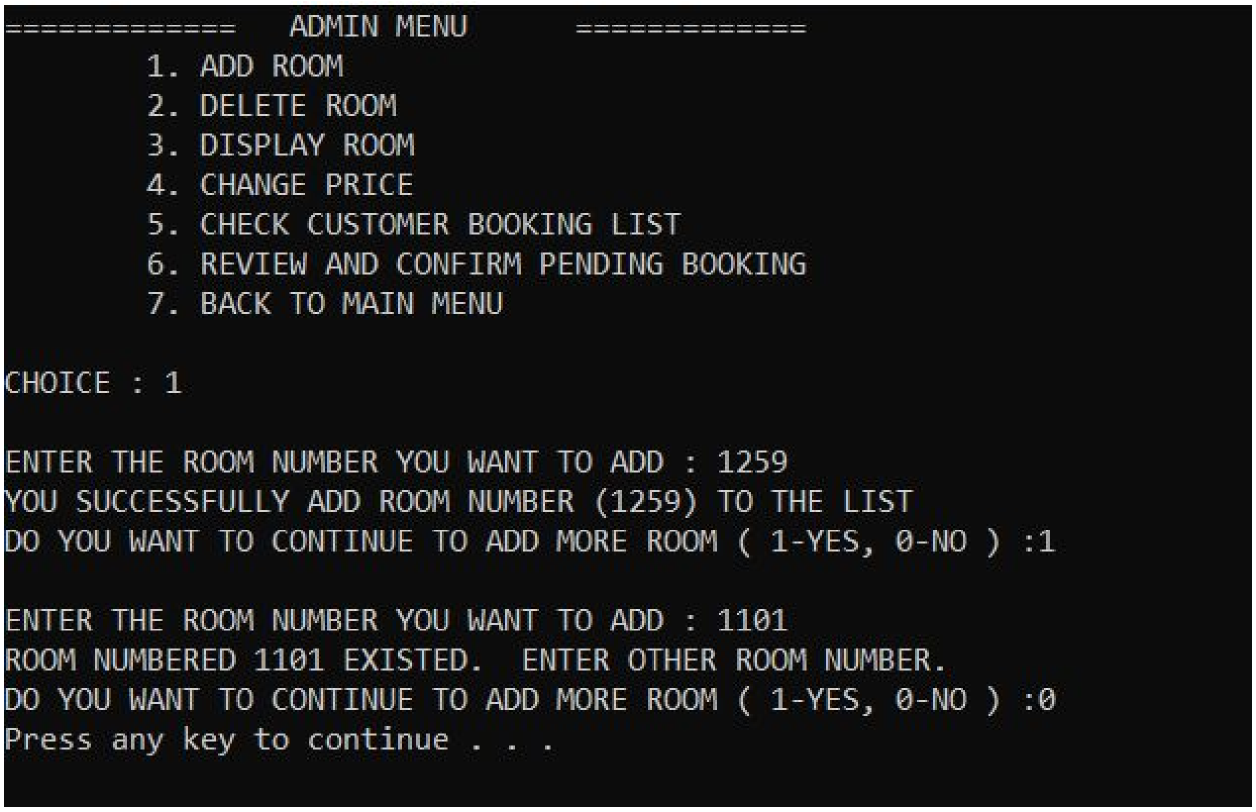


```
=============   ADMIN MENU      =============
        1. ADD ROOM
        2. DELETE ROOM
        3. DISPLAY ROOM
        4. CHANGE PRICE
        5. CHECK CUSTOMER BOOKING LIST
        6. REVIEW AND CONFIRM PENDING BOOKING
        7. BACK TO MAIN MENU

CHOICE :
```

Screen 2: Admin Menu

Screen 2: If the user enters integer 1 after choosing admin, it will display an admin menu.The user needs to enter the integer value in the range of 1 to 7 according to what choices they want. If the user enters the other number, the system will prompt invalid choice and the screen will display again after the user presses any key.

Prepared By: ONG YIN REN

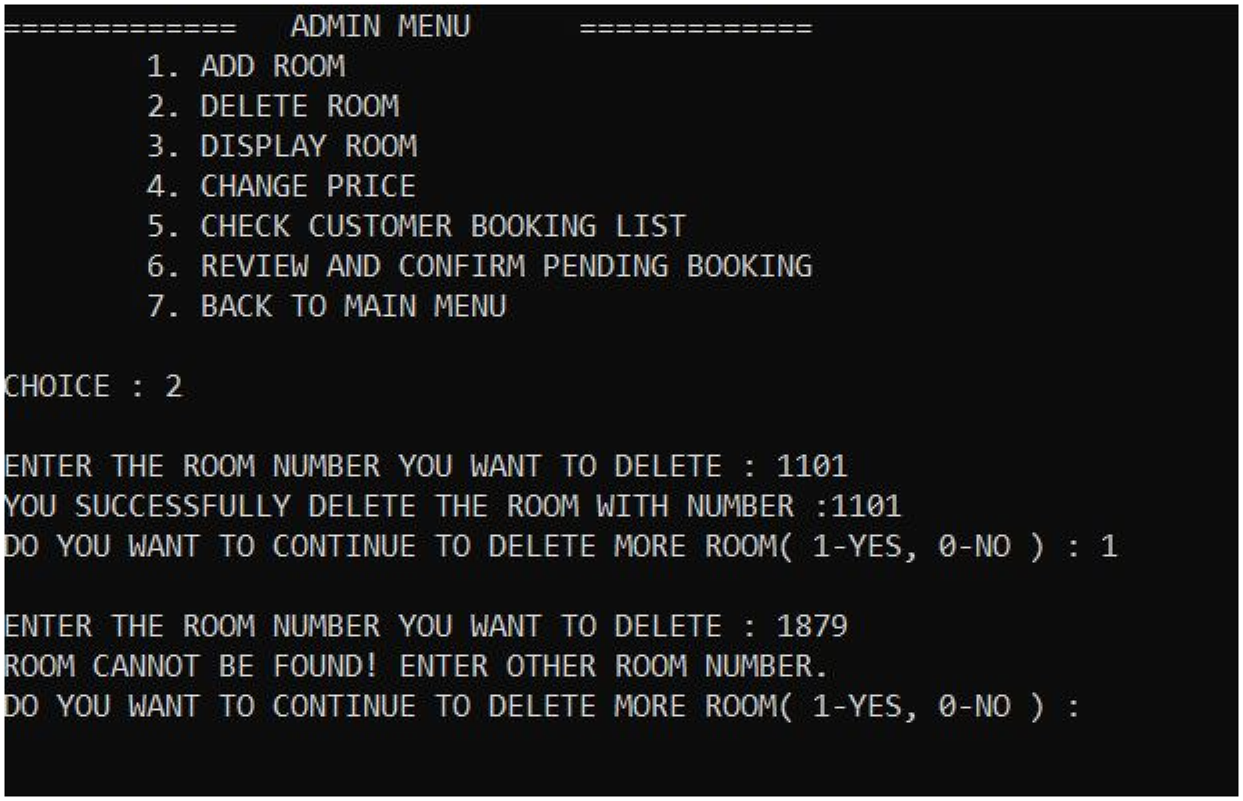


```
=============   ADMIN MENU      =============
        1. ADD ROOM
        2. DELETE ROOM
        3. DISPLAY ROOM
        4. CHANGE PRICE
        5. CHECK CUSTOMER BOOKING LIST
        6. REVIEW AND CONFIRM PENDING BOOKING
        7. BACK TO MAIN MENU

CHOICE : 1

ENTER THE ROOM NUMBER YOU WANT TO ADD : 1259
YOU SUCCESSFULLY ADD ROOM NUMBER (1259) TO THE LIST
DO YOU WANT TO CONTINUE TO ADD MORE ROOM ( 1-YES, 0-NO ) :1

ENTER THE ROOM NUMBER YOU WANT TO ADD : 1101
ROOM NUMBERED 1101 EXISTED.  ENTER OTHER ROOM NUMBER.
DO YOU WANT TO CONTINUE TO ADD MORE ROOM ( 1-YES, 0-NO ) :0
Press any key to continue . . .
```

Screen 3: Admin - Adding room

Screen 3: For choice 1, if the user enters a new room number, it will display a successful message. If the user enters an existing room number, it will display an error message and request the user to try another room number. After that, it will display an option to have or not for continuing the adding room process. If wanted, enter 1 else enter 0 if not interested to continue adding. The screen of the admin menu will display again after the user presses any key.

Prepared By: CHIAM WOOI CHIN

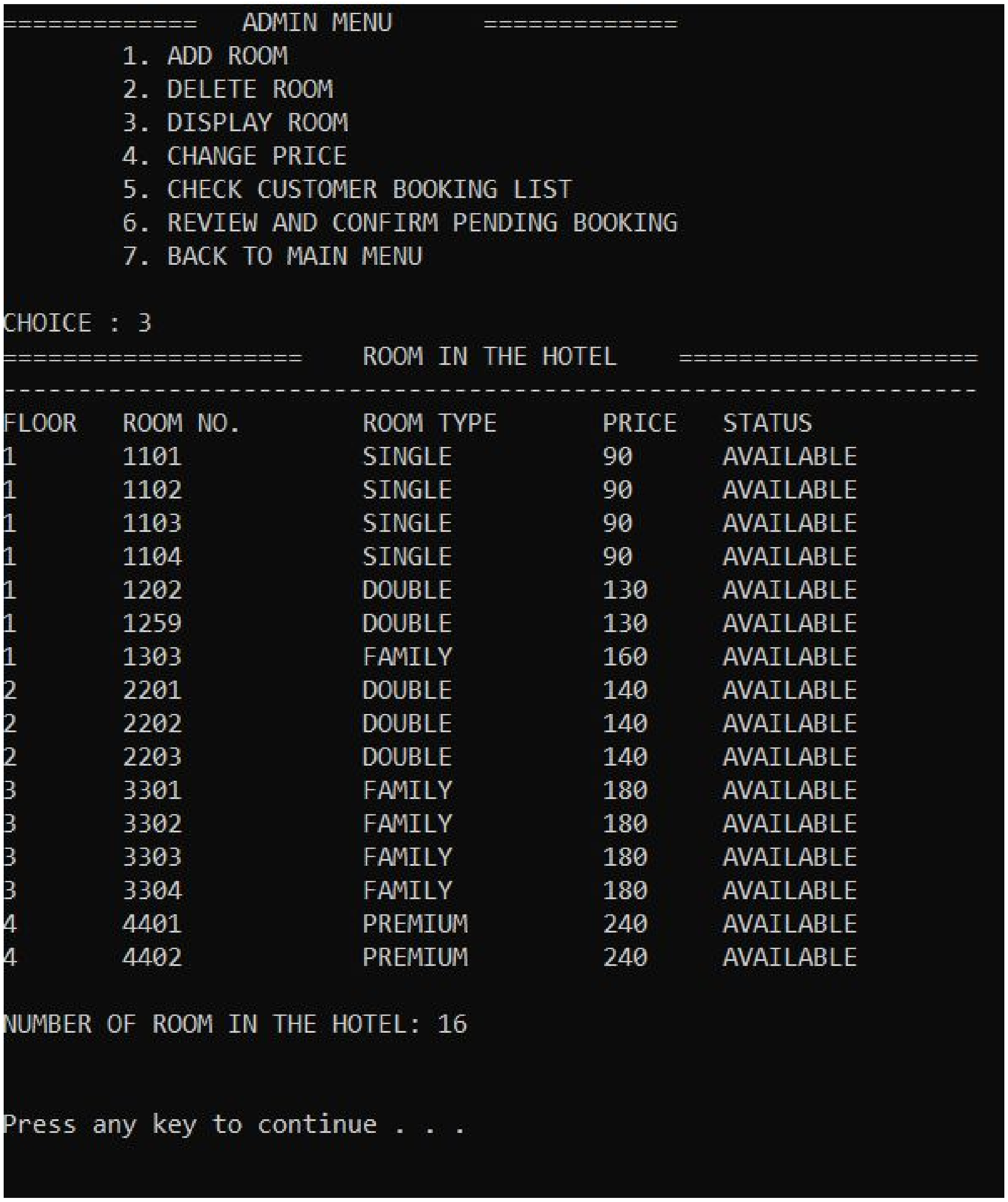


```
============   ADMIN MENU      ============
        1. ADD ROOM
        2. DELETE ROOM
        3. DISPLAY ROOM
        4. CHANGE PRICE
        5. CHECK CUSTOMER BOOKING LIST
        6. REVIEW AND CONFIRM PENDING BOOKING
        7. BACK TO MAIN MENU

CHOICE : 2

ENTER THE ROOM NUMBER YOU WANT TO DELETE : 1101
YOU SUCCESSFULLY DELETE THE ROOM WITH NUMBER :1101
DO YOU WANT TO CONTINUE TO DELETE MORE ROOM( 1-YES, 0-NO ) : 1

ENTER THE ROOM NUMBER YOU WANT TO DELETE : 1879
ROOM CANNOT BE FOUND! ENTER OTHER ROOM NUMBER.
DO YOU WANT TO CONTINUE TO DELETE MORE ROOM( 1-YES, 0-NO ) :
```

Screen 4: Admin - Deleting room

Screen 4: For choice 2, it will allow the user to enter a room number to delete a room. If the user enters an existing room number, it will display a successful message or else it will request the user to enter another room number. After that, it will display an option to have or not for continuing the deleting room process. If wanted, enter 1 else enter 0 if not interested to continue deleting. The screen of the admin menu will display again after the user presses any key.

Prepared By: CHIAM WOOI CHIN

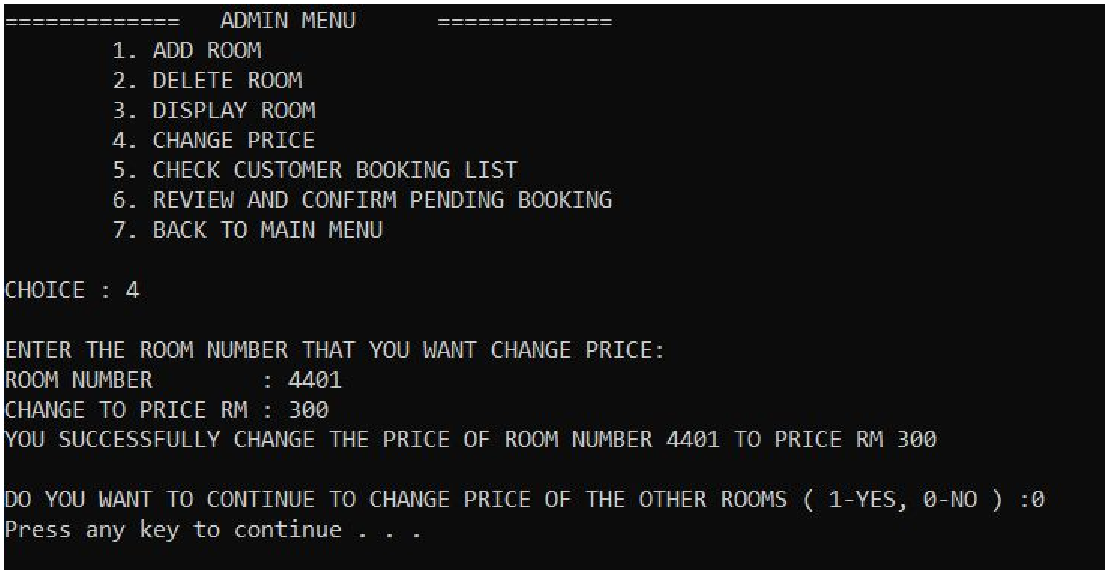


```
=============   ADMIN MENU      =============
        1. ADD ROOM
        2. DELETE ROOM
        3. DISPLAY ROOM
        4. CHANGE PRICE
        5. CHECK CUSTOMER BOOKING LIST
        6. REVIEW AND CONFIRM PENDING BOOKING
        7. BACK TO MAIN MENU

CHOICE : 3
====================    ROOM IN THE HOTEL    ====================
-----------------------------------------------------------------
FLOOR   ROOM NO.        ROOM TYPE       PRICE   STATUS
1       1101            SINGLE          90      AVAILABLE
1       1102            SINGLE          90      AVAILABLE
1       1103            SINGLE          90      AVAILABLE
1       1104            SINGLE          90      AVAILABLE
1       1202            DOUBLE          130     AVAILABLE
1       1259            DOUBLE          130     AVAILABLE
1       1303            FAMILY          160     AVAILABLE
2       2201            DOUBLE          140     AVAILABLE
2       2202            DOUBLE          140     AVAILABLE
2       2203            DOUBLE          140     AVAILABLE
3       3301            FAMILY          180     AVAILABLE
3       3302            FAMILY          180     AVAILABLE
3       3303            FAMILY          180     AVAILABLE
3       3304            FAMILY          180     AVAILABLE
4       4401            PREMIUM         240     AVAILABLE
4       4402            PREMIUM         240     AVAILABLE

NUMBER OF ROOM IN THE HOTEL: 16


Press any key to continue . . .
```

Screen 5: Admin - Displaying rooms in the hotel

Screen 5: For choice 3 of admin, it will display the rooms in the hotel and their status. The details of the room in the hotel included floor, room number, room type, price and status of availability. If the user enters any key, the system will prompt to the admin menu.

Prepared By: CHIAM WOOI CHIN

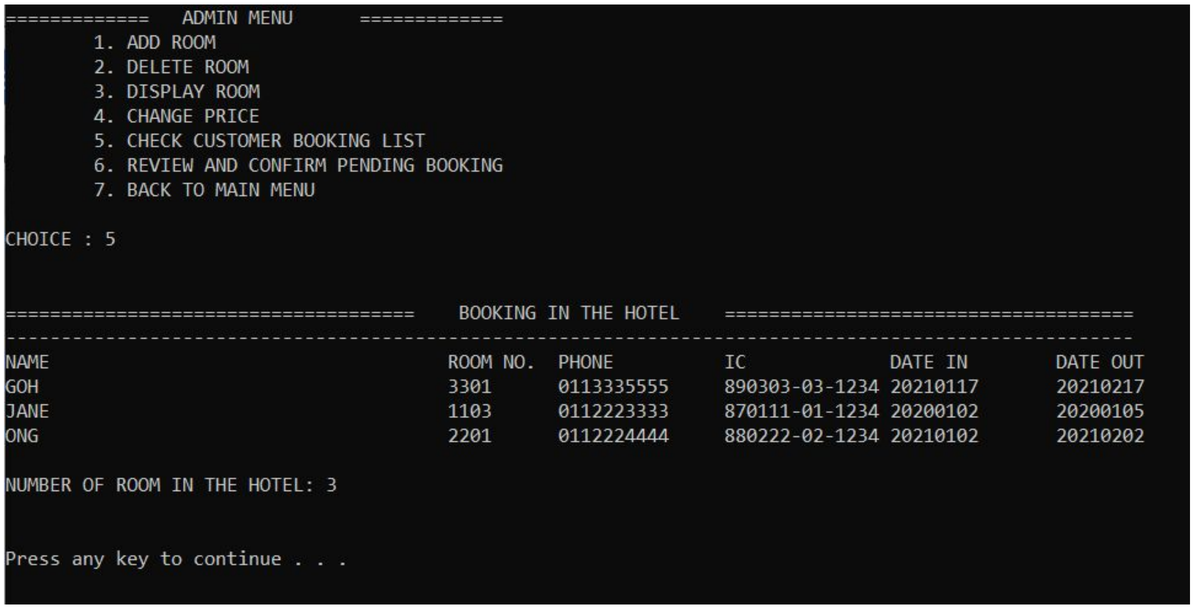


```
=============   ADMIN MENU      =============
        1. ADD ROOM
        2. DELETE ROOM
        3. DISPLAY ROOM
        4. CHANGE PRICE
        5. CHECK CUSTOMER BOOKING LIST
        6. REVIEW AND CONFIRM PENDING BOOKING
        7. BACK TO MAIN MENU

CHOICE : 4

ENTER THE ROOM NUMBER THAT YOU WANT CHANGE PRICE:
ROOM NUMBER        : 4401
CHANGE TO PRICE RM : 300
YOU SUCCESSFULLY CHANGE THE PRICE OF ROOM NUMBER 4401 TO PRICE RM 300

DO YOU WANT TO CONTINUE TO CHANGE PRICE OF THE OTHER ROOMS ( 1-YES, 0-NO ) :0
Press any key to continue . . .
```

Screen 6: Admin - Change price of the rooms

Screen 6: User needs to insert a room number that wants to change price. After that, the user needs to insert a new price for the related room number. Then, the system will display a successful message on the screen. Users need to enter 0 or 1 to continue or not for changing the room price.

Prepared By: NG JING ER

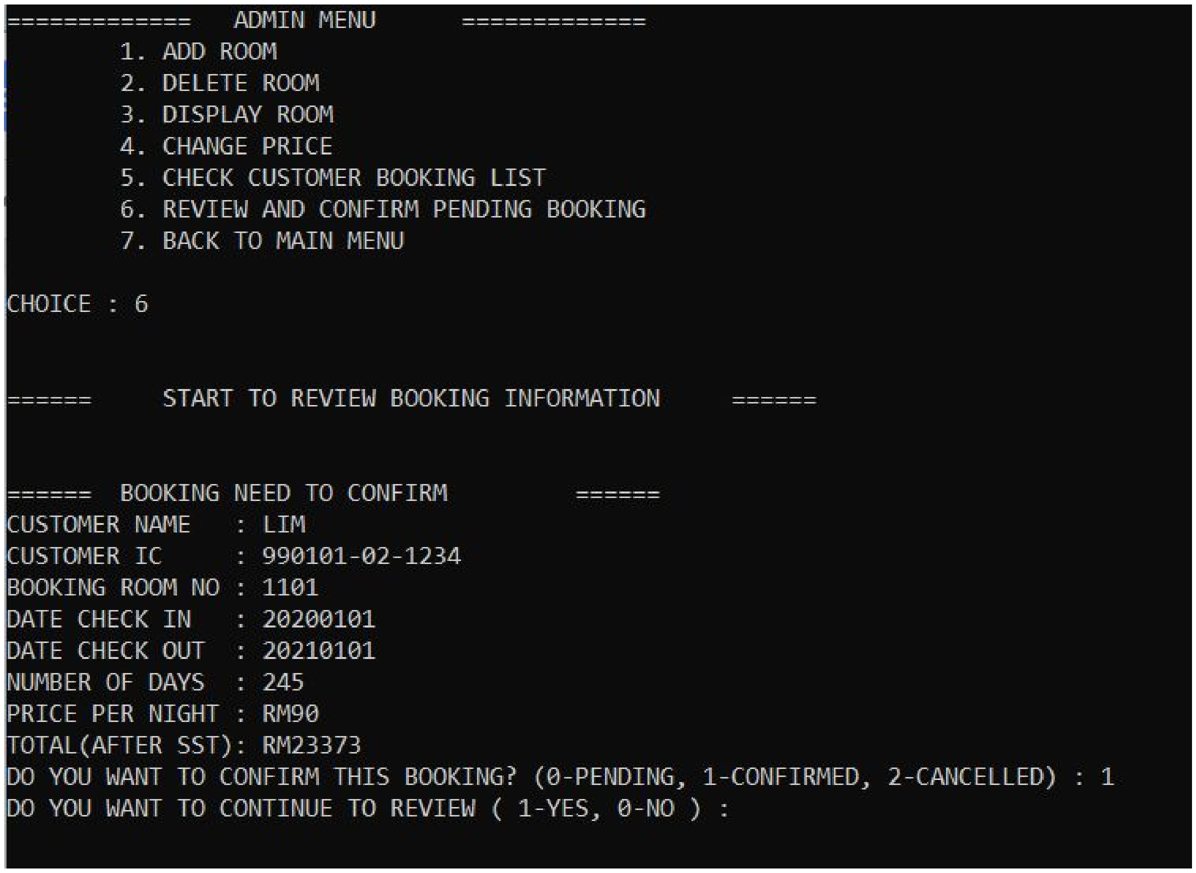


```
=============   ADMIN MENU      =============
        1. ADD ROOM
        2. DELETE ROOM
        3. DISPLAY ROOM
        4. CHANGE PRICE
        5. CHECK CUSTOMER BOOKING LIST
        6. REVIEW AND CONFIRM PENDING BOOKING
        7. BACK TO MAIN MENU

CHOICE : 5


====================================    BOOKING IN THE HOTEL    ====================================
----------------------------------------------------------------------------------------------------
NAME                                     ROOM NO.  PHONE          IC             DATE IN        DATE OUT
GOH                                      3301      0113335555     890303-03-1234 20210117       20210217
JANE                                     1103      0112223333     870111-01-1234 20200102       20200105
ONG                                      2201      0112224444     880222-02-1234 20210102       20210202

NUMBER OF ROOM IN THE HOTEL: 3


Press any key to continue . . .
```

Screen 7: Admin - Checking booking list

Screen 7: The system will show all the booking details of the customer. Admin able to check with the existing booking and the customer details. Users need to enter any key to continue.

Prepared By: NG JING ER

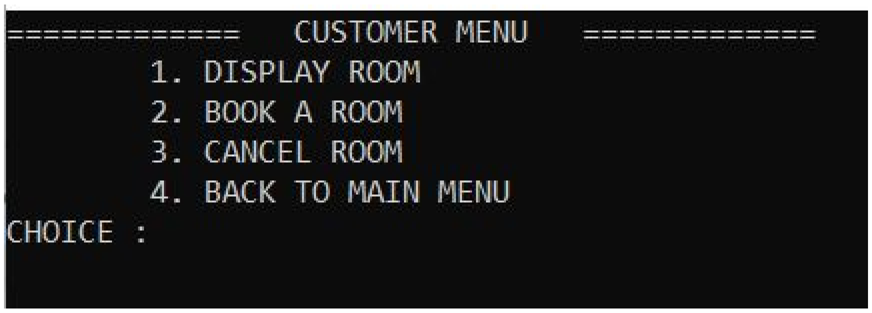


```
=============   ADMIN MENU      =============
        1. ADD ROOM
        2. DELETE ROOM
        3. DISPLAY ROOM
        4. CHANGE PRICE
        5. CHECK CUSTOMER BOOKING LIST
        6. REVIEW AND CONFIRM PENDING BOOKING
        7. BACK TO MAIN MENU

CHOICE : 6


======     START TO REVIEW BOOKING INFORMATION     ======


======  BOOKING NEED TO CONFIRM         ======
CUSTOMER NAME   : LIM
CUSTOMER IC     : 990101-02-1234
BOOKING ROOM NO : 1101
DATE CHECK IN   : 20200101
DATE CHECK OUT  : 20210101
NUMBER OF DAYS  : 245
PRICE PER NIGHT : RM90
TOTAL(AFTER SST): RM23373
DO YOU WANT TO CONFIRM THIS BOOKING? (0-PENDING, 1-CONFIRMED, 2-CANCELLED) : 1
DO YOU WANT TO CONTINUE TO REVIEW ( 1-YES, 0-NO ) :
```

Screen 8: Admin - Reviewing and confirm booking

Screen 8: The system will show all the bookings that need to be confirmed. To change the status of booking, the user needs to enter the integer value in the range of 0 to 2 where 0 is pending, 1 is confirmed and 2 is cancelled.

Prepared By; NG JING ER

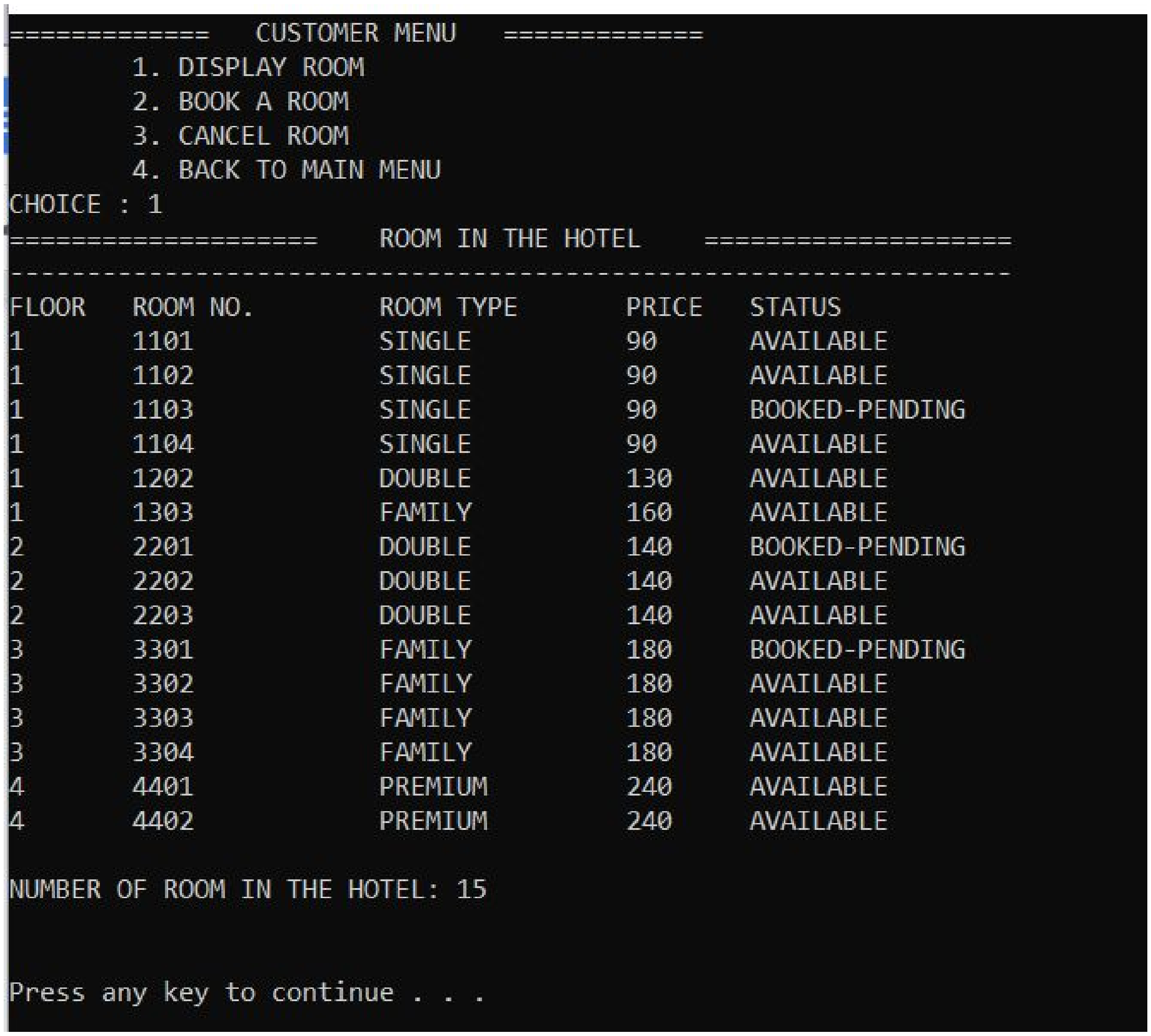


```
=============   CUSTOMER MENU   =============
        1. DISPLAY ROOM
        2. BOOK A ROOM
        3. CANCEL ROOM
        4. BACK TO MAIN MENU
CHOICE :
```

Screen 9: Customer Menu

Screen 9: The user needs to enter the integer value in the range of 1 to 4 according to what choices they want. If the user enters the other number, the system will prompt invalid choice and the screen will display again after the user presses any key.

Prepared By: ONG YIN REN

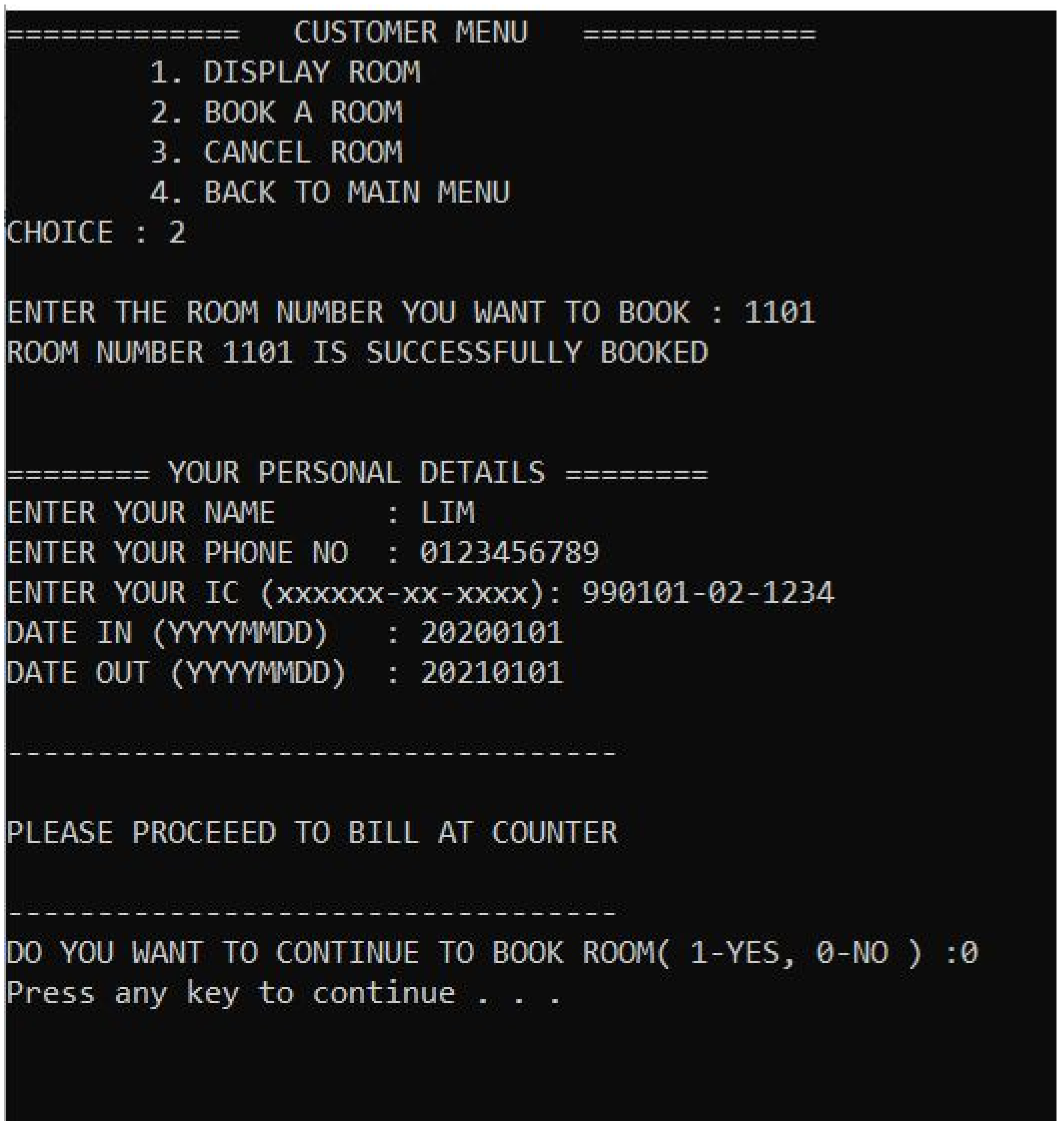


```
=============   CUSTOMER MENU   =============
        1. DISPLAY ROOM
        2. BOOK A ROOM
        3. CANCEL ROOM
        4. BACK TO MAIN MENU
CHOICE : 1
====================    ROOM IN THE HOTEL    ====================
-----------------------------------------------------------------
FLOOR   ROOM NO.        ROOM TYPE       PRICE   STATUS
1       1101            SINGLE          90      AVAILABLE
1       1102            SINGLE          90      AVAILABLE
1       1103            SINGLE          90      BOOKED-PENDING
1       1104            SINGLE          90      AVAILABLE
1       1202            DOUBLE          130     AVAILABLE
1       1303            FAMILY          160     AVAILABLE
2       2201            DOUBLE          140     BOOKED-PENDING
2       2202            DOUBLE          140     AVAILABLE
2       2203            DOUBLE          140     AVAILABLE
3       3301            FAMILY          180     BOOKED-PENDING
3       3302            FAMILY          180     AVAILABLE
3       3303            FAMILY          180     AVAILABLE
3       3304            FAMILY          180     AVAILABLE
4       4401            PREMIUM         240     AVAILABLE
4       4402            PREMIUM         240     AVAILABLE

NUMBER OF ROOM IN THE HOTEL: 15


Press any key to continue . . .
```

Screen 10: Customer - Displaying room in the hotel

Screen 10: It will display all the rooms in the hotel. The details of the room in the hotel included floor, room number, room type, price and status. If the user enters any key, the system will prompt to the customer menu.

Prepared By: GOH JO EY

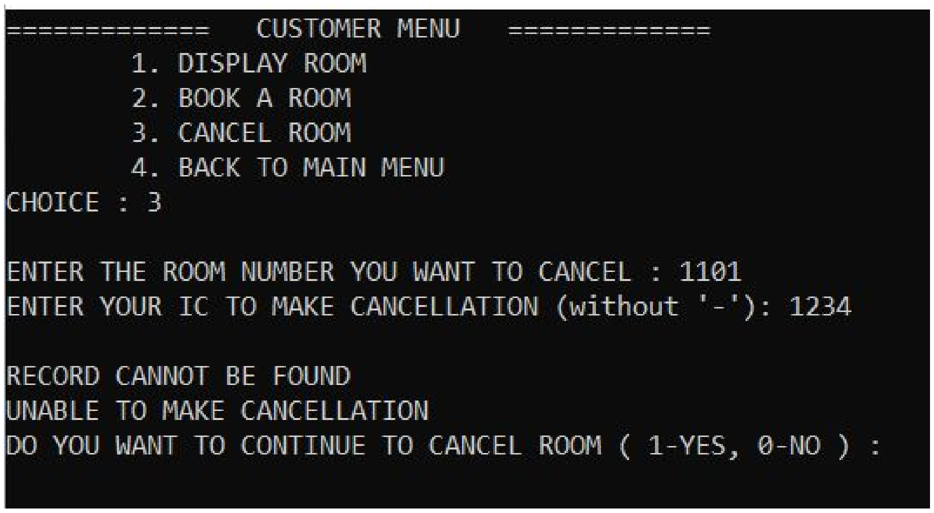


```
=============   CUSTOMER MENU   =============
        1. DISPLAY ROOM
        2. BOOK A ROOM
        3. CANCEL ROOM
        4. BACK TO MAIN MENU
CHOICE : 2

ENTER THE ROOM NUMBER YOU WANT TO BOOK : 1101
ROOM NUMBER 1101 IS SUCCESSFULLY BOOKED


======== YOUR PERSONAL DETAILS ========
ENTER YOUR NAME      : LIM
ENTER YOUR PHONE NO  : 0123456789
ENTER YOUR IC (xxxxxx-xx-xxxx): 990101-02-1234
DATE IN (YYYYMMDD)   : 20200101
DATE OUT (YYYYMMDD)  : 20210101

----------------------------------

PLEASE PROCEEED TO BILL AT COUNTER

----------------------------------
DO YOU WANT TO CONTINUE TO BOOK ROOM( 1-YES, 0-NO ) :0
Press any key to continue . . .
```

Screen 11: Customer - Booking a room

Screen 11: It will display a successfully booked message if the user entered the matched room number in the hotel. After that, users need to enter their personality such as their name, phone number, number of ic, date check in and date check out with the correct format.Then, it will display “please proceed to the bill at the counter”. Users will then need to enter 1 or 0 to continue or not for their booking.

Prepared By: GOH JO EY

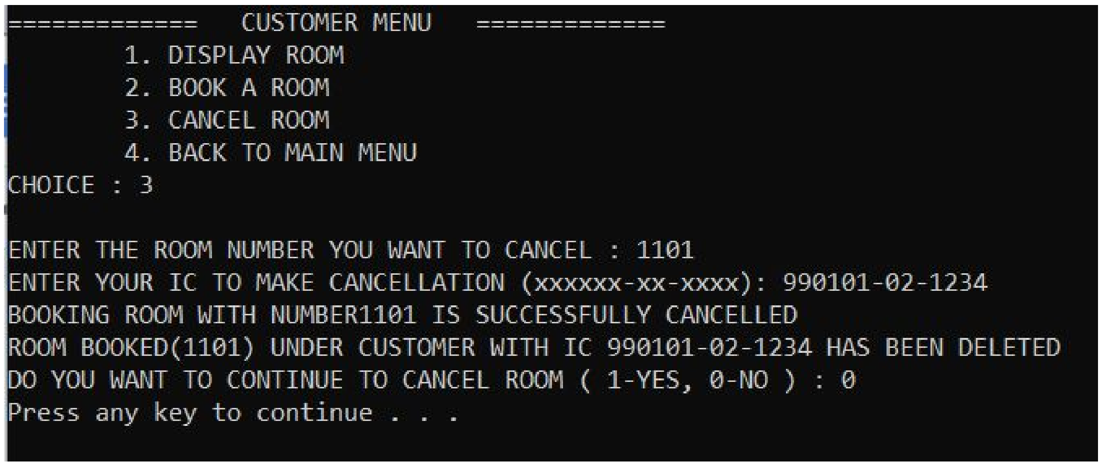


```
=============   CUSTOMER MENU   =============
        1. DISPLAY ROOM
        2. BOOK A ROOM
        3. CANCEL ROOM
        4. BACK TO MAIN MENU
CHOICE : 3

ENTER THE ROOM NUMBER YOU WANT TO CANCEL : 1101
ENTER YOUR IC TO MAKE CANCELLATION (without '-'): 1234

RECORD CANNOT BE FOUND
UNABLE TO MAKE CANCELLATION
DO YOU WANT TO CONTINUE TO CANCEL ROOM ( 1-YES, 0-NO ) :
```

Screen 12: Customer UNABLE cancelling the room

Screen 12: If the user enters the room that wishes to cancel is invalid, then the system will display a record not found and unable to make cancellation. After that, the user needs to enter 1 for continuing or 0 quit the cancellation process.

Prepared By: GOH JO EY



```
=============   CUSTOMER MENU   =============
        1. DISPLAY ROOM
        2. BOOK A ROOM
        3. CANCEL ROOM
        4. BACK TO MAIN MENU
CHOICE : 3

ENTER THE ROOM NUMBER YOU WANT TO CANCEL : 1101
ENTER YOUR IC TO MAKE CANCELLATION (xxxxxx-xx-xxxx): 990101-02-1234
BOOKING ROOM WITH NUMBER1101 IS SUCCESSFULLY CANCELLED
ROOM BOOKED(1101) UNDER CUSTOMER WITH IC 990101-02-1234 HAS BEEN DELETED
DO YOU WANT TO CONTINUE TO CANCEL ROOM ( 1-YES, 0-NO ) : 0
Press any key to continue . . .
```

Screen 13: Customer SUCCESSFULLY cancelling the room

Screen 13: User needs to enter the room that wishes to cancel is valid, then the system will display successfully cancelled. After that, the user needs to enter 1 to continue or 0 to stop the cancellation process.

Prepared By: GOH JO EY

PART 4: SYSTEM TESTING

1. Test Case Matrix for the Book Room By Customer Use Case

	Scenario	Room No.	Customer Name	Phone No.	IC	Date in (YYYYMMDD)	Date out (YYYYMMDD)	Expected Result	Test Result
1.	Successfully Booking	4401	Jane	0123456789	870821-00-0000	20201231	20210131	Successfully booked message displayed and request for continue	Successfully booked message displayed and request for continue
2.	Successfully Booking	3301	WC Chiam	0129876543	990821011234	20210213	20210214	Successfully booked message displayed and request for continue	Successfully booked message displayed and request for continue
3.	Unidentified room number	1000	N/A	N/A	N/A	N/A	N/A	Error message displayed and request for continue	Wrong room number

2. Test Case Matrix for the Cancel Room By Customer Use Case

	Scenario	Room No.	IC (xxxxxx-xx-xxxx)	Expected Result	Test Result
1.	Successfully cancelling the room	1101	870821-00-0000	Booking room is successfully cancelled	Booking room with room number under customer with IC has been deleted.
2.	Unidentified room number	1234	N/A	Error messages and back to customer menu	Error: Record cannot be found unable to make cancellation
3.	Unidentified IC	1101	123456	Error messages and back to customer menu	Error: Record cannot be found unable to make cancellation

3. Test Case Matrix for the Update Room By Admin Use Case

	Scenario	**Room No.**	**Expected Result**	**Test Result**
1.	Successfully add a new room	1108	Successfully added message displayed and request for continue	Successfully added message displayed and request for continue
	Adding an existing room	1101	Error message displayed and request for continue	Error: The room exists.
2.	Successfully delete a room	1101	Successfully added message displayed and request for continue	Successfully deleted message displayed and request for continue
3.	Unidentified room number to be delete	5501	Error message displayed and request for continue	Error: The room number cannot be found

4. Test Case Matrix for the Change Room Price By Admin Use Case

	Scenario	**Room No.**	**Price (RM)**	**Expected Result**	**Test Result**
1.	Successfully changed a room price	1101	100	Successfully changed message displayed and request for continue	Successfully changed message displayed and request for continue
2.	Unidentified room number	5501	400	Error message displayed and request for continue	Error: The room number cannot be found

5. Test Case Matrix for the Check Room Availability Use Case

	Scenario	Expected Result	Test Result
1.	Successfully checked room availability	Floor number, room number, room type, room price and room availability are displayed.	Floor number, room number, room type, room price and room availability are displayed.

6. Test Case Matrix for the Check Customer Booking Use Case

	Scenario	Expected Result	Test Result
1.	Successfully checked customer booking list	Customer name, room no., phone no., IC, date in, date out and number of rooms are displayed.	Customer name, room no., phone no., IC, date in, date out and number of rooms are displayed.
2.	No customer book the room	No record of customer booking list	No record of customer booking list

7. Test Case Matrix for the Review and Confirm Pending Booking By Admin Use Case

	Scenario	Status	Expected Result	Test Result
1.	Review and confirm the pending booking (0- PENDING 1- CONFIRMED 2- CANCELLED)	1	Customer name, IC, booking room no., date check in, date check out, number of days, room price per night and total room price are displayed. The status of room availability changed.	Customer name, IC, booking room no., date check in, date check out, number of days, room price per night and total room price are displayed. The status of room availability changed.
2.	No pending booking	N/A	No record of customer booking information	No record of customer booking information

PART 5: DEVELOPMENT ACTIVITIES

Meeting Date	Members Participate in the meeting	Activity	Task for each member	Task Achieved (yes/No)
05/01/2021	1. Chiam Wooi Chin 2. Goh Jo Ey 3. Ng Jing Er 4. Ong Yin Ren	Discuss the title, objective and concepts of the system.	All members do the research for the hotel booking system related and the implementation of data structure in the system.	1. Yes 2. Yes 3. Yes 4. Yes
08/01/2021	1. Chiam Wooi Chin 2. Goh Jo Ey 3. Ng Jing Er 4. Ong Yin Ren	Discuss the system analysis, system requirement, use case and system design.	1. System Design (flowchart 1-4) 2. System Design (flowchart 1-4) 3. System Design (flowchart 5-8) 4. System Design (flowchart 5-8)	1. Yes 2. Yes 3. Yes 4. Yes
13/01/2021	1. Chiam Wooi Chin 2. Goh Jo Ey 3. Ng Jing Er 4. Ong Yin Ren	Do the system design by programming using C++ and complete report.	1. Admin choice 1,2,3 2. Customer choice 1,2 3. Admin choice 4,5,6 4. Customer choice, complete report	1. Yes 2. Yes 3. Yes 4. Yes
16/01/2021 - 28/01/2021	1. Chiam Wooi Chin 2. Goh Jo Ey 3. Ng Jing Er 4. Ong Yin Ren	Discuss and improve the system design and C++ coding and complete report.	All members improve source code, complete report and prepare for presentation video	1. Yes 2. Yes 3. Yes 4. Yes

PART 6 : APPENDIX HARDCOPY OF SOURCE CODE

```
/*
GROUP MINI PROJECT- HOTEL BOOKING SYSTEM
GROUP MEMBERS:
 1. GOH JO EY        A19EC0047
 2. ONG YIN REN      A19EC0204
 3. CHIAM WOOI CHIN  A19EC0034
 4. NG JING ER       A19EC0115

*/

#include <iostream>
#include <iomanip>
#include <string>
#include <conio.h>
#include <stdlib.h>
#include <cmath>
#include <cctype>

using namespace std;

class billNode{
        public:
        int roomNum;
        float price;
        string ic;
        string name;
        double total;
        string DateIn;
        string DateOut;
        int days;
        billNode *next;
};


class billQueue{
        billNode *frontPtr,*backPtr;
        public:
                billQueue(){
                        backPtr=NULL;
                        frontPtr=NULL;
                }
                ~billQueue(){
                        billNode *temp = frontPtr;
                        while (temp){
                        frontPtr = temp->next;
                        delete temp;
                        temp=frontPtr;
                        }
                }
                bool isEmpty(){
```

```
		return (backPtr == NULL && frontPtr == NULL);
	}

	//implementation of queue
	void enQueue(int rn,float p,string ic,string n,string in,string
out,int d){
		billNode *temp=new billNode;
		if (backPtr == NULL) {
		temp->next = NULL;
		temp->roomNum = rn;
		temp->price = p;
		temp->ic = ic;
		temp->name= n;
		temp->DateIn = in;
			temp->DateOut = out;
			temp->days=d;
			frontPtr = backPtr = temp;
		}
		else {
		backPtr->next = temp;
		temp->roomNum = rn;
		temp->price = p;
		temp->ic = ic;
		temp->name= n;
		temp->DateIn = in;
			temp->DateOut = out;
			temp->days=d;
		temp->next = NULL;
		backPtr = temp;
		}
	}
	void displayInvoice(billNode *temp){
		if(temp->days<=1){
			temp->days==1;
		}
		temp->total=temp->price*1.06*(temp->days*1.0);
		cout<<"\n\n======       BOOKING NEED TO CONFIRM
======";
		cout<<"\nCUSTOMER NAME\t: "<<temp->name<<endl;
		cout<<"CUSTOMER IC\t: "<<temp->ic<<endl;
		cout<<"BOOKING ROOM NO\t: "<<temp->roomNum<<endl;
		cout<<"DATE CHECK IN\t: "<<temp->DateIn<<endl;
		cout<<"DATE CHECK OUT\t: "<<temp->DateOut<<endl;
		cout<<"NUMBER OF DAYS\t: "<<temp->days<<endl;
		cout<<"PRICE PER NIGHT\t: RM"<<temp->price<<endl;
		cout<<"TOTAL(AFTER SST): RM"<<temp->total<<endl;
	}

	//implementation of queue
	string deQueue(){
```

```
                        billNode *temp=new billNode;
                        temp = frontPtr;
                        if (frontPtr == NULL) {
                        cout<<"NO PENDING BOOKING NEED TO BE CONFIRMED"<<endl;
                                return "NULL";
                        }
                        else if (temp->next != NULL) {

                                int choice;
                                displayInvoice(temp);
                                do{
                                cout<<"DO YOU WANT TO CONFIRM THIS BOOKING?
(0-PENDING, 1-CONFIRMED, 2-CANCELLED) : ";
                                cin>>choice;
                                        if(choice==1){
                                                temp = temp->next;
                                        frontPtr = temp;
                                        return "BOOKED-CONFIRMED";
                                }
                                else if(choice==0){
                                temp = temp->next;
                                frontPtr = temp;
                                        return "BOOKED-PENDING";
                                        }
                                        else if(choice==2){
                                        temp = temp->next;
                                frontPtr = temp;
                                        return "CANCELLED";
                                        }
                                }while(choice<0&&choice>2);


                        }
                        else {
                                int choice;
                                displayInvoice(temp);
                                do{
                                cout<<"\n\nDO YOU WANT TO CONFIRM THIS BOOKING?
(0-PENDING, 1-CONFIRMED, 2-CANCELLED) : ";
                                cin>>choice;
                                        if(choice==1){
                                        temp = temp->next;
                                frontPtr = temp;
                                return "BOOKED-CONFIRMED";
                                }
                                else if(choice==0){
                                return "BOOKED-PENDING";
                                        }
                                        else if(choice==2){
                                        return "CANCELLED";
                                        }
                                }while(choice<0&&choice>2);
                        }

                }
                billNode* getFrontRN(){
                        billNode *t=new billNode;
                                if(frontPtr!=NULL){
                                t=frontPtr;
                                }
                        return t;
                }
};

class custNode{
        public:
        string custName;
        string custPhone;
        string ic;
        string DateIn;
        string DateOut;
        int roomNum;
        custNode *next;
};

class customer{
        custNode *head;
        public:
                customer(){head=NULL;}

                void sortedCustIn(string n, string p, string ic, string in,string
out,int rn){
                        int currIndex=0;
                        custNode *currNode=head;
                        custNode *prevNode=NULL;
                                while(currNode && n > currNode->custName){
                                        prevNode=currNode;
                                        currNode=currNode->next;
                                        currIndex++;
                                }
                        custNode *newNode=new custNode;
                        newNode->custName=n;
                        newNode->custPhone=p;
                        newNode->ic=ic;
                        newNode->DateIn=in;
                        newNode->DateOut=out;
                        newNode->roomNum=rn;
                                if (currIndex==0){
                                        newNode->next=head;
                                        head=newNode;
```

```
				}
				else{
				newNode->next=prevNode->next;
				prevNode->next=newNode;
				}
		}
		void deleteCust(string ic,int rn){
			custNode* prevNode = NULL;
			custNode* currNode = head;
			int currIndex = 1;

			string IC=currNode->ic.substr(0,12);
			int y = atoi(IC.c_str());
			int x= atoi(ic.c_str());

			while ((currNode && x!= y)||(currNode &&
currNode->roomNum!=rn)) {
				prevNode = currNode;
				currNode = currNode->next;
				currIndex++;
			}

			if(x==y){
				if (currNode) {
					if (prevNode) {
						prevNode->next = currNode->next;
						delete currNode;
					}
					else {
						head = currNode->next;
						delete currNode;
					}
				}
			cout<<"ROOM BOOKED("<<rn<<") UNDER CUSTOMER WITH IC
"<<ic<<" HAS BEEN DELETED "<<endl;
			}
		else{
		cout<<"BOOKING RECORD OF "<<rn<<" UNDER CUSTOMER WITH IC "<<ic<<"
ICANNOT BE FOUND"<<endl;
			}
		}

		bool findCust(string _ic){
			custNode* currNode = head;
			int currIndex = 1;
			string IC=currNode->ic.substr(0,13);
			int y = atoi(IC.c_str());
			int x= atoi(_ic.c_str());
			if (x==y)
				return true;
```

```
                        else
                                return false;
                }

                void displayCustRoom(){
                        int num = 0;
                        custNode* currNode = head;
                        cout<<"\n\n=====================================
BOOKING IN THE HOTEL    ======================================"<<endl;

cout<<"---------------------------------------------------------------------------
----------------------------"<<endl;
                        cout<<left<<setw(40) <<"NAME"<<setw(11)<<"ROOM
NO."<<setw(15)<<"PHONE"<<setw(20)<<"IC"<<setw(12)<<"DATE IN"<<setw(12)<<"DATE
OUT"<<endl;

                        while (currNode != NULL){
                                cout <<left<<setw(40)<<currNode->custName
                                         <<setw(11)<< left<<currNode->roomNum
                                         <<setw(15)<< left<<currNode->custPhone
                                         <<setw(20)<< left<<currNode->ic
                                         <<setw(12)<< left<<currNode->DateIn
                                         <<setw(12)<< left<<currNode->DateOut
                                         << endl;
                                currNode = currNode->next;
                                num++;
                        }
                        cout << "\nNUMBER OF ROOM IN THE HOTEL: " << num <<
endl<<endl<<endl;
                }

};

class roomNode{
        public:
        int floorNo;
        int roomNum;
        int roomtype;
        float price;
        int status;
        roomNode *next;
        roomNode *prev;
};

string rt(int rt){
        if(rt==1){
                return "SINGLE";
        }
        else if(rt==2){
                return "DOUBLE";
        }
        else if(rt==3){
                return "FAMILY";
        }
        else if(rt==4){
                return "PREMIUM";
        }
        else{
                return "DEFAULT";
        }
}

string status(int status){
        if(status==1){
                return "BOOKED-PENDING";
        }
        else if(status==2){
                return "BOOKED-CONFIRMED";
        }
        else if(status==0){
                return "AVAILABLE";
        }
        else{
                return "DEFAULT";
        }
}

class room{
        private:
                roomNode *head;

        public:
                room(){
                        head=NULL;
                }

                void insertSortedRoom(int roomNum){
                        int currIndex=0;
                        roomNode *currNode=head;
                        roomNode *prevNode=NULL;
                                while(currNode && roomNum > currNode->roomNum){
                                        prevNode=currNode;
                                        currNode=currNode->next;
                                        currIndex++;
                                }
                        roomNode *newNode=new roomNode;
                        newNode->roomNum=roomNum;
                        newNode->floorNo=roomNum/1000;
                        newNode->roomtype=(roomNum-(roomNum/1000*1000))/100;
                        newNode->price=setprice(newNode);
        newNode->status=0;
        if (currIndex==0){
                newNode->next=head;
                head=newNode;
        }
        else{
                newNode->next=prevNode->next;
                prevNode->next=newNode;
        }
}
float setprice(roomNode *n){
        float p=0;
        roomNode *newNode=new roomNode;
        newNode=n;
        if(newNode->roomtype==1){
                p+=80.00;
        }
        else if(newNode->roomtype==2){
                p+=120.00;
        }
        else if(newNode->roomtype==3){
                p+=150.00;
        }
        else if(newNode->roomtype==4){
                p+=200.00;
        }
        else{
                p=0.00;
        }
        p+=(newNode->floorNo*10.00);
        return p;
}

void deleteRoom(int rn){
        roomNode* prevNode = NULL;
        roomNode* currNode = head;
        int currIndex = 1;
        while (currNode && currNode->roomNum != rn) {
                prevNode = currNode;
                currNode = currNode->next;
                currIndex++;
        }
        if (currNode) {
        if (prevNode) {
                prevNode->next = currNode->next;
                delete currNode;
        }
        else {
                head = currNode->next;
                delete currNode;
```

```
                        }
                        cout<<"YOU SUCCESSFULLY DELETE THE ROOM WITH NUMBER
:"<<rn<<endl;
                }
                else{
                cout<<"ROOM NUMBER ("<<rn<<") DOES NOT EXIST."<<endl;
                        }
                }
                bool bookRoom(int rn){
                        roomNode *newNode=new roomNode;
                        newNode=findRoom(rn);
                        if(!newNode){
                                cout<<"ROOM NUMBER "<<rn<<" CANNOT BE
FOUND\n"<<endl;
                                return false;
                        }
                        else{
                                if(newNode->status==0){
                                        newNode->status=1;
                                        cout<<"ROOM NUMBER "<<rn<<" IS SUCCESSFULLY
BOOKED\n\n"<<endl;
                                        return true;
                                }
                                else{
                                        cout<<"ROOM NUMBER "<<rn<<" IS NOT
AVAILABLE\n\n"<<endl;
                                        return false;
                                }
                        }
                }

                bool cancelRoom(int rn){
                        roomNode *newNode=new roomNode;
                        newNode=findRoom(rn);
                        if(!newNode){
                                cout<<"ROOM NUMBER "<<rn<<" CANNOT BE FOUND"<<endl;
                                return false;
                        }
                        else{
                                if(newNode->status==1){
                                        newNode->status=0;
                                        cout<<"BOOKING ROOM WITH NUMBER"<<rn<<" IS
SUCCESSFULLY CANCELLED"<<endl;
                                        return true;
                                }
                                else{
                                        cout<<"BOOKING ROOM NUMBER ("<<rn<<")
CANNOT BE FOUND"<<endl;
                                        cout<<"UNABLE TO MAKE CANCELLATION"<<endl;
                                        return false;
                        }
                }
        };

                bool findBookRoom(int rn){
                        roomNode *newNode=new roomNode;
                        newNode=findRoom(rn);
                        if(!newNode){
                                return false;
                        }
                        else{
                                if(newNode->status==1){
                                        return true;
                                }
                                else{
                                        return false;
                                }
                        }
                }

                roomNode* findRoom(int rn){
                        roomNode* currNode = head;
                        int currIndex = 1;
                        while (currNode && currNode->roomNum != rn) {
                        currNode = currNode->next;
                        currIndex++;
                        }
                        if (currNode)
                                return currNode;
                        else
                        return 0;
                }

                void displayRoom(){
                        int num = 0;
                        roomNode* currNode = head;
                        cout<<"====================    ROOM IN THE HOTEL
===================="<<endl;

cout<<"-----------------------------------------------------------------"<<endl;
                        cout<<"FLOOR\tROOM NO.\tROOM TYPE\tPRICE\tSTATUS"<<endl;


                        while (currNode != NULL){
                                cout <<currNode->floorNo<<"\t"
                                         <<currNode->roomNum <<"\t\t"
                                         <<rt(currNode->roomtype)<<"\t\t"
                                         <<currNode->price<<"\t"
                                         <<status(currNode->status)<<"\t"
                                         << endl;
                                        currNode = currNode->next;
                                        num++;
                                }
                                cout << "\nNUMBER OF ROOM IN THE HOTEL: " << num <<
endl<<endl<<endl;
                        }
};

int getDays(string date) {
        int year = atoi(date.substr(0, 4).c_str());
        int month = atoi(date.substr(4, 2).c_str());
        int day = atoi(date.substr(6, 2).c_str());
        int ans = 0;
        for (int i = 1900; i < year; ++ i) {
            if (((year % 4 == 0) && (year % 100 != 0)) || (year % 400 == 0)) {
                ans += 366;
            } else {
                ans += 365;
            }
        }
        for (int i = 1; i < month; ++ i) {
            switch(i) {
                case 1: ans += 31; break;
                case 2: ans += (((year % 4 == 0) && (year % 100 != 0)) || (year %
400 == 0)) ? 29 : 28; break;
                case 3: ans += 31; break;
                case 4: ans += 30; break;
                case 5: ans += 31; break;
                case 6: ans += 30; break;
                case 7: ans += 31; break;
                case 8: ans += 31; break;
                case 9: ans += 30; break;
                case 10: ans += 31; break;
                case 11: ans += 30; break;
                case 12: ans += 31; break;
            }
        }
        return ans += day - 1;
    }

int daysBetweenDates(string date1, string date2) {
        return abs(getDays(date1) - getDays(date2));
}

int main(){

        //CREATE HOTEL ROOM
        room h;
        customer cn;
        billQueue bq;
        int choice;

        //ADD DEFAULT ROOM
        int 
defaultroom[]={1101,1202,1303,1102,1103,1104,2201,2202,2203,3301,3302,3303,3304,440
1,4402} ;
        for(int x=0;x<15;x++){
                h.insertSortedRoom(defaultroom[x]);
        }

        do{
                system("CLS");
                cout<<"============            HOTEL BOOKING SYSTEM 
============"<<endl;
                cout<<"\t\t\tLOGIN AS"<<endl;
                cout<<"\t\t\t1. ADMIN\n\t\t\t2. CUSTOMER \n\t\t\t3. EXIT\n"<<endl;
                cout<<"CHOICE : ";
                cin>>choice;
                switch(choice){
                        case 1:{
                                int ch1;
                                int rn;
                                int cont1;
                                do{
                                system("CLS");
                                cout<<"============    ADMIN MENU 
============"<<endl;
                                cout<<"\t1. ADD ROOM\n\t2. DELETE ROOM\n\t3. 
DISPLAY ROOM\n\t4. CHANGE PRICE\n\t5. CHECK CUSTOMER BOOKING LIST\n\t6. REVIEW AND 
CONFIRM PENDING BOOKING\n\t7. BACK TO MAIN MENU\n"<<endl;
                                cout<<"CHOICE : ";
                                cin>>ch1;
                                switch(ch1){
                                case 1:
                                        cont1=1;
                                        do{
                                        cout<<"\nENTER THE ROOM NUMBER YOU WANT TO 
ADD : ";
                                        cin>>rn;
                                        if(!h.findRoom(rn)){
                                                h.insertSortedRoom(rn);
                                                cout<<"YOU SUCCESSFULLY ADD ROOM 
NUMBER ("<<rn<<") TO THE LIST"<<endl;
                                        }
                                        else{
                                                cout<<"ROOM NUMBERED "<<rn<<" 
EXISTED.  ENTER OTHER ROOM NUMBER."<<endl;
                                        }
                                        cout<<"DO YOU WANT TO CONTINUE TO ADD MORE 
ROOM ( 1-YES, 0-NO ) :";
```

```
						cin>>cont1;
						}while(cont1!=0);
						break;
				case 2:
						cont1=1;
						do{
						cout<<"\nENTER THE ROOM NUMBER YOU WANT TO 
DELETE : ";
						cin>>rn;
						if(h.findRoom(rn)){
							h.deleteRoom(rn);
						}
						else{
							cout<<"ROOM CANNOT BE FOUND! ENTER 
OTHER ROOM NUMBER."<<endl;
						}
						cout<<"DO YOU WANT TO CONTINUE TO DELETE 
MORE ROOM( 1-YES, 0-NO ) : ";
						cin>>cont1;
						}while(cont1!=0);
						break;
				case 3:
						h.displayRoom();
						break;
				case 4:
						cont1=1;
						do{
						cout<<"\nENTER THE ROOM NUMBER THAT YOU 
WANT CHANGE PRICE: "<<endl;
						cout<<"ROOM NUMBER        : ";
						cin>>rn;
						if(h.findRoom(rn)){
							float p;
							cout<<"CHANGE TO PRICE RM : ";
							cin>>p;
							roomNode *newNode=h.findRoom(rn);
							newNode->price=p;
							cout<<"YOU SUCCESSFULLY CHANGE THE 
PRICE OF ROOM NUMBER "<<rn<<" TO PRICE RM "<<p<<endl<<endl;
						}
						else{
							cout<<"ROOM NUMBER ("<<rn<<") 
CANNOT BE FOUND. ENTER OTHER ROOM NUMBER."<<endl<<endl;
						}
						cout<<"DO YOU WANT TO CONTINUE TO CHANGE 
PRICE OF THE OTHER ROOMS ( 1-YES, 0-NO ) :";
						cin>>cont1;
						}while(cont1!=0);
						break;
				case 5:
```

```
                                        cn.displayCustRoom();
                                        break;
                                case 6:
                                        cont1=1;

                                        cout<<"\n\n======     START TO REVIEW
BOOKING INFORMATION     ======\n";
                                        do{

                                        billNode* rno=new billNode;
                                        rno=bq.getFrontRN();
                                        string con=bq.deQueue();
                                        roomNode *t=new roomNode;
                                        t=h.findRoom(rno->roomNum);
                                        if(con=="BOOKED & CONFIRMED"){
                                                t->status=2;

                                                break;
                                        }
                                        else if(con=="CANCELLED"){
                                                t->status=0;
                                                break;
                                        }
                                        else if(con=="BOOKED & PENDING"){
                                                t->status=1;

bq.enQueue(rno->roomNum,rno->price,rno->ic,rno->name,rno->DateIn,rno->DateOut,rno->
days);
                                                break;
                                        }
                                        else {}

                                        cout<<"DO YOU WANT TO CONTINUE TO REVIEW (
1-YES, 0-NO ) :";
                                        cin>>cont1;

                                }while(cont1!=0);
                                        break;
                                case 7:
                                        break;
                                default:
                                        cout<<"INVALID CHOICE."<<endl;
                                        break;
                        }
                                system("pause");
                                getch();

                        }while(ch1!=7);
                        break;
                        }
                        break;

                        case 2:{
                                int ch2;
                                int rn,rn2;
                                int cont2;
                                do{
                                system("CLS");
                                cout<<"============    CUSTOMER MENU
============"<<endl;
                                cout<<"\t1. DISPLAY ROOM\n\t2. BOOK A ROOM\n\t3.
CANCEL ROOM\n\t4. BACK TO MAIN MENU"<<endl;
                                cout<<"CHOICE : ";
                                cin>>ch2;
                                switch(ch2){
                                case 1:
                                        h.displayRoom();
                                        break;
                                case 2:
                                        cont2=1;
                                        do{
                                        cout<<"\nENTER THE ROOM NUMBER YOU WANT TO
BOOK : ";

                                        cin>>rn;
                                        bool book=h.bookRoom(rn);
                                        if(book){
                                        string n, p, ic, in, out;
                                        int din,min,yin,dout,mout,yout;
                                        char s;
                                        cout<<"======== YOUR PERSONAL DETAILS
========"<<endl;
                                        cout<<"ENTER YOUR NAME      : ";
                                        cin.ignore();
                                        getline(cin,n);
                                        cout<<"ENTER YOUR PHONE NO  : ";
                                        getline(cin,p);
                                        cout<<"ENTER YOUR IC (xxxxxx-xx-xxxx): ";
                                        getline(cin,ic);
                                        cout<<"DATE IN (YYYYMMDD)   : ";
                                        getline(cin,in);
                                        cout<<"DATE OUT (YYYYMMDD)  : ";
                                        getline(cin,out);
                                        int days=daysBetweenDates(in,out);
                                        cn.sortedCustIn(n,p,ic,in,out,rn);

cout<<"\n----------------------------------\n";
                                        cout<<"\nPLEASE PROCEEED TO BILL AT
COUNTER\n";

cout<<"\n----------------------------------\n";
```

```
                                        roomNode *t=new roomNode;
                                        t=h.findRoom(rn);
                                        bq.enQueue(rn,t->price,ic,n,in,out,days);
                                        }
                                        cout<<"DO YOU WANT TO CONTINUE TO BOOK 
ROOM( 1-YES, 0-NO ) :";
                                        cin>>cont2;
                                        }while(cont2!=0);
                                        break;

                                case 3:
                                        cont2=1;
                                        do{
                                        string ic2;
                                        int rn2=0;
                                        cout<<"\nENTER THE ROOM NUMBER YOU WANT TO 
CANCEL : ";
                                        cin>>rn2;
                                        cout<<"ENTER YOUR IC TO MAKE CANCELLATION 
(xxxxxx-xx-xxxx): ";
                                        cin.ignore();
                                        getline(cin,ic2);

                                        bool targetRoom=h.findBookRoom(rn2);
                                        if(targetRoom){
                                                bool targetCust=cn.findCust(ic2);
                                                if(targetCust){
                                                        h.cancelRoom(rn2);
                                                        cn.deleteCust(ic2,rn2);
                                                }
                                                else{
                                                cout<<"\nRECORD CANNOT BE 
FOUND\nUNABLE TO MAKE CANCELLATION"<<endl;
                                                }
                                        }
                                        else{
                                                cout<<"RECORD CANNOT BE 
FOUND\nUNABLE TO MAKE CANCELLATION"<<endl;
                                        }

                                        cout<<"DO YOU WANT TO CONTINUE TO CANCEL 
ROOM ( 1-YES, 0-NO ) : ";
                                        cin>>cont2;
                                        }while(cont2!=0);
                                        break;
                                case 4:
                                        break;
                                default:
                                        cout<<"INVALID CHOICE."<<endl;
                                        break;
```

```
                        }
                                system("pause");
                                getch();

                        }while(ch2!=4);
                                break;
                        }
                        case 3:
                                cout<<"THANK YOU! BYE BYE."<<endl;
                                exit(0);
                        default:
                                cout<<"INVALID CHOICE."<<endl;
                                break;
                }

        getch();
        }while (choice!=3);

        return 0;
}
```

APPENDIX V: PEER REVIEW ASSESSMENT

Name: **Chiam Wooi Chin A19EC0034**																									
Team member name:	Cooperative (1-min, 5-max)					Hardworking (1-min, 5-max)					Punctuality (1-min, 5-max)					Knowledge Sharing (1-min, 5-max)					Good personality (1-min, 5-max)				
Goh Jo Ey	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]
Ng Jing Er	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]
Ong Yin Ren	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]
TOTAL	**15**					**15**					**15**					**15**					**15**				

Name: **Goh Jo Ey A19EC0047**																									
Team member name:	Cooperative (1-min, 5-max)					Hardworking (1-min, 5-max)					Punctuality (1-min, 5-max)					Knowledge Sharing (1-min, 5-max)					Good personality (1-min, 5-max)				
Chiam Wooi Chin	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]
Ng Jing Er	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]
Ong Yin Ren	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]
TOTAL	**15**					**15**					**15**					**15**					**15**				

Name: **Ng Jing Er A19EC0115**																									
Team member name:	Cooperative (1-min, 5-max)					Hardworking (1-min, 5-max)					Punctuality (1-min, 5-max)					Knowledge Sharing (1-min, 5-max)					Good personality (1-min, 5-max)				
Goh Jo Ey	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]
Chiam Wooi Chin	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]
Ong Yin Ren	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]
TOTAL	**15**					**15**					**15**					**15**					**15**				

Name: **Ong Yin Ren A19EC0204**																									
Team member name:	Cooperative (1-min, 5-max)					Hardworking (1-min, 5-max)					Punctuality (1-min, 5-max)					Knowledge Sharing (1-min, 5-max)					Good personality (1-min, 5-max)				
Goh Jo Ey	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]
Ng Jing Er	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]
Chiam Wooi Chin	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]	1	2	3	4	[✓ 5]
TOTAL	**15**					**15**					**15**					**15**					**15**				