UTM
UNIVERSITI TEKNOLOGI MALAYSIA

Department of Computer Science
Faculty of Computing

GROUP PROJECT

FINAL PROJECT

STACK, QUEUE AND TREE

Programme	:	Bachelor of Computer Science (*Computer Network and Security*)
Subject Code	:	SECJ2013
Subject Name	:	DATA STRUCTURE AND ALGORITHMS
Session-Sem	:	2023/2024-1

Prepared by	:	1) Aidan Andrew Daniel (A22EC0036) 2) Muhammad Nabil Iman bin Mohd Fazli (A22EC0046) 3) Muhammad Haziq bin Saamsol (A22EC0214)
Section	:	03
Group	:	GOATS
Lecturer	:	Mrs Lizawati binti Mi Yusuf

Table of Content

Problem analysis

The efficient operation of a hospital depends on the efficient maintenance of patient records. The obstacles of efficient patient record keeping in a hospital context are addressed by the Hospital Management System. By implementing necessary data structures concepts which are stack, queue and tree, the system seeks to manage patient records effectively and guarantee a dynamic and well-organized workflow inside the hospital.

One necessity that is particularly important is patient records management, which includes efficient monitoring of prescriptions, check-ins, and appointments. The selected data structure or structures must be implemented with precision and efficiency, and they must work in unison with the unique requirements of the hospital administration system.

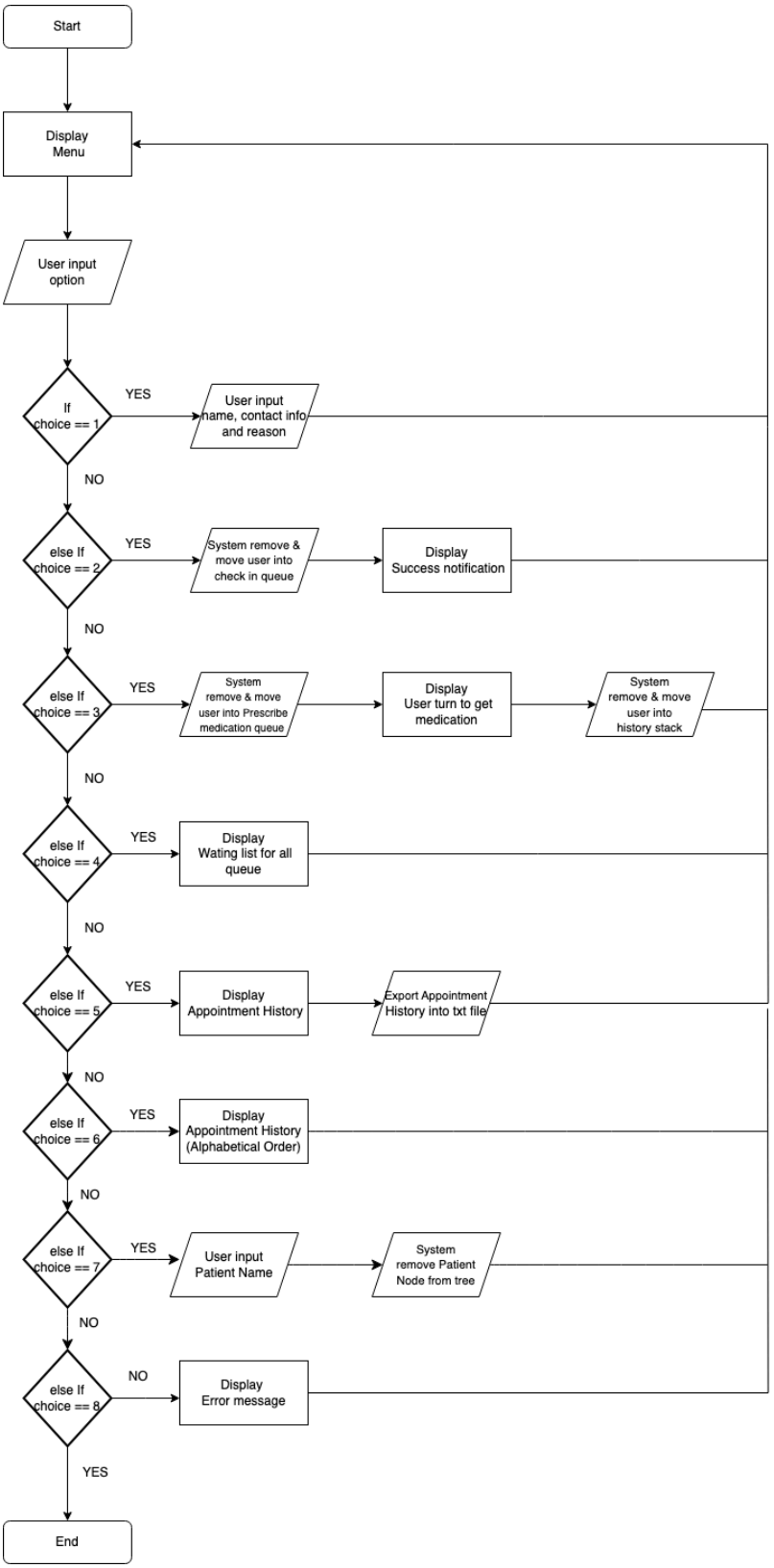
The main objectives are to create a reliable software programme for managing hospitals, implement the right data structures to manage patient records, make use of the stack, queue, and tree data structure principles and ensure that C++ is used correctly to apply the data structures concepts.

Requests for prescriptions, check-ins, and appointments will be handled using a queue-based data structure. The queue guarantees a systematic and well-organized procedure and is implemented using arrays or linked lists. So, this system will keep track of daily appointments by using a stack data structure that follows the concept of first in last out.

Among the challenges the system will confront include properly allocating waiting numbers for appointments, ensuring that the check-in procedure is arranged based on waiting numbers, and implementing a memory-clearing mechanism when the system goes down. The data structure idea that will be used will be implemented to tackle the problems that the system is experiencing.

Design

- **Flowchart**



Data structure Implementation

Queue : Appointment system

The implementation of queues using linked lists ensures an effective way to manage medication prescriptions, check-ins, and appointments. The Hospital Reservation System is designed to handle patient appointments, check-ins, and prescriptions. Queue functions are implemented in the system to handle appointments requests, patients checking-in and medicines prescription in the system. Patients are able to check in, make appointment requests and get prescriptions filled, doctors are added to the system so that patients can consult with them. The system uses a queue-based data structure to monitor prescriptions, check-ins, and patient appointments. Every queue is implemented as a linked list, where each node corresponds to a particular patient or physician. Patients who request an appointment are given a waiting number, and lines are kept orderly and organised using the queue function. The queue system also made sure that only the patient with the next expected waiting number are able to check-in, ensuring a correct and maintained order of the queue system.

Stack : Appointment History

In this hospital management system, we only improve part of the hospital functionality which is the appointment system . It also employs a linked list to offer an infinite size to hold many patients in a day of appointment operation. This stack will automatically add patients who have completed their visits and received their prescribed medicine into it. Furthermore, the stack follows the concept of the Last in First Out (LIFO) principle, which states that the most recent patient added to the stack is the first to be popped. With this stack, we also include a function that generates a report of the appointment history, including all patient information. This report will be generated before the system is closed down. After the report is generated the stack element will pop one by one until empty to clear the memory.

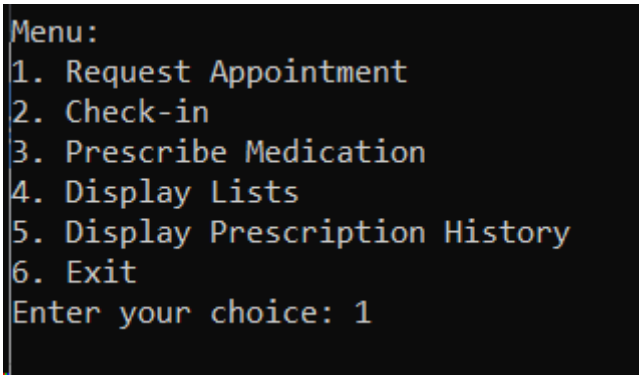
Tree : Appointment History (Alphabetical Order)

In this system that we have created, we give emphasis on arranging the patient history by using the concept of Tree implementation. To be more specific the usage of the Binary Tree concept. The tree set in the code has a root and nodes where each have two children, labelled as left and right child. In BST, all values of the left subtree must be a smaller value set by the system while those in the right subtree are larger values set by the system. Therefore, the structure of the tree highly depends on the first node that is inserted to the tree. The insertion function inserts new nodes to the tree, recursively finding the right position while comparing the value set by system (in our case : name) of the new node to its current subtree. The In-order traversing method traverses the tree in the manner of in-order throughout the whole tree. So, while traversing, we have made it so that it displays the current node it is in - in this case sorted by name. There is also a node deletion mechanism that removes nodes when specified the name. There are three scenarios which the code has supported for which are deleting a node with no children, one child or two children.

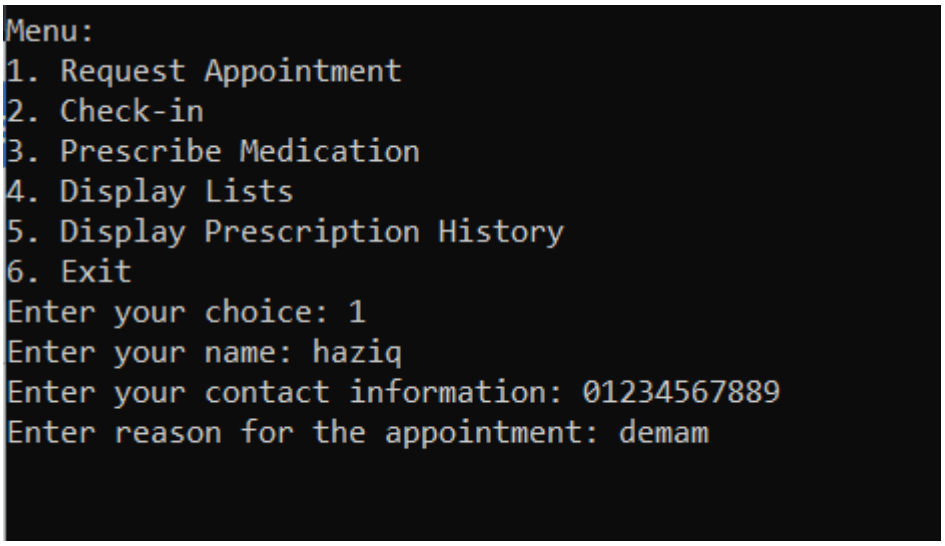
User guideline

This system focuses on part of the hospital's function. So we Focus on the appointment system. The system is easy to learn and easy to use with one click all operation is completed. Almost all operations are automated by the system.

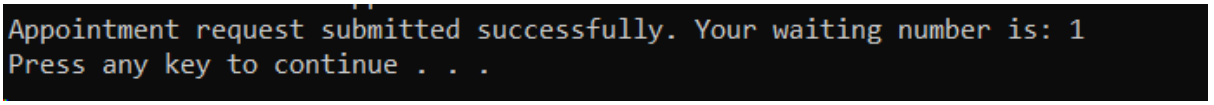
Make Appointment



1. The system will display a menu.
2. Choose (key in 1) to add a new appointment.



3. Insert Patient name, contact information, and reason for appointment.
4. Press enter



5. The system will display a success notification that means that the appointment has been added.

Check-in for an appointment.

```
Menu:
1. Request Appointment
2. Check-in
3. Prescribe Medication
4. Display Lists
5. Display Prescription History
6. Exit
Enter your choice: 2
```

1. The system will display a menu.
2. Choose (key in 2) option 2 to check in for an appointment (Check means that the patient gets ready to make an appointment).

```
Check-in successful. You are now in the waiting list.
Press any key to continue . . .
```

3. The system will display a success message to notify that the patient has a successful check-in.
4. The patient will be in a waiting queue for the next check-up.

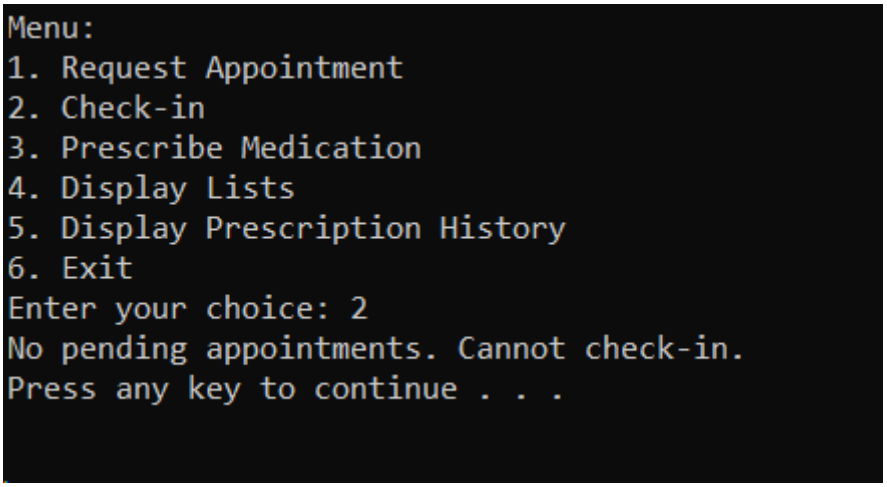
```
Menu:
1. Request Appointment
2. Check-in
3. Prescribe Medication
4. Display Lists
5. Display Prescription History
6. Exit
Enter your choice: 4

Not checked-in List:
Waiting Number: 1        Name: haziq     Contact: 1234567889     Reason: demam

Checked-in List:
Empty

Prescription waiting List:
Empty
Press any key to continue . . .
```

5. You can check if the user is in the queue by keying in option 4 in the menu. (you do this with other operations to check if the user is added to the queue or not)



6. If you got this message means that there is no patient in the queue so cannot do the operation.

Call the patient to get a Medication prescription

```
Menu:
1. Request Appointment
2. Check-in
3. Prescribe Medication
4. Display Lists
5. Display Prescription History
6. Exit
Enter your choice: 3
```

1. The system will display a menu.
2. Choose (key in 3) option 3 to move the patient in the queue to get medication after the doctor's discussion.
3. The system will automatically remove patients that have got their medication.

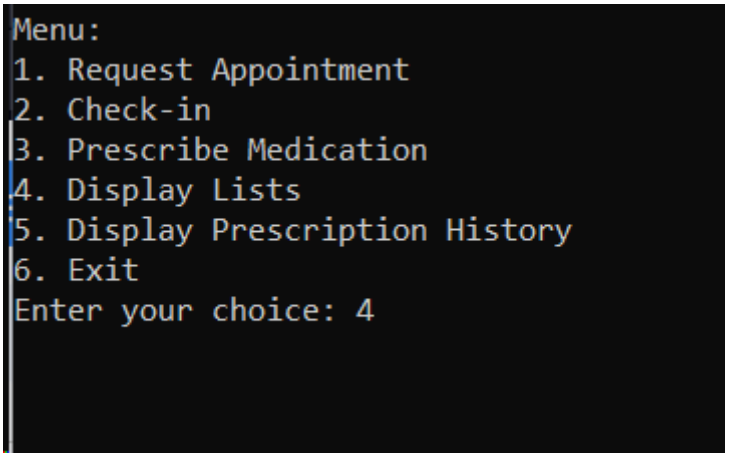
```
Menu:
1. Request Appointment
2. Check-in
3. Prescribe Medication
4. Display Lists
5. Display Prescription History
6. Exit
Enter your choice: 3
Prescription requested for patient haziq.
Medication prescribed for patient haziq.
Press any key to continue . . .
```

4. The system will display a success message of the patient getting a turn to get medication and the medication that the patient is given.

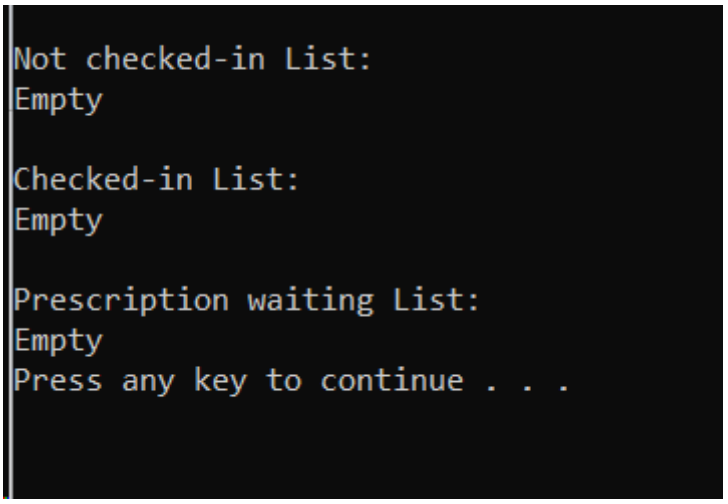
```
Menu:
1. Request Appointment
2. Check-in
3. Prescribe Medication
4. Display Lists
5. Display Prescription History
6. Exit
Enter your choice: 3
No patients in the waiting list. Cannot prescribe medication.
Press any key to continue . . .
```

5. If you got this message means that there is no patient in the queue so cannot do the operation.

Check the waiting list queue



1. The system will display a menu.
2. Choose (key in 4) option 4 to look at all the waiting queues.



currently no patient in the list.

3. The system will display all the waiting queues for patients that have not checked in yet, have checked in, and are waiting to get medication.

To view the history of the Hospital appointment

```
Menu:
1. Request Appointment
2. Check-in
3. Prescribe Medication
4. Display Lists
5. Display Prescription History
6. Exit
Enter your choice: 5
```

1. The system will display a menu.
2. Choose (key in 5) option 5 to view the list of all patients that do a discussion with the doctor.

```
Prescription History:
Patient Name: haziq      Contact: 01234567889    Reason: demam
Press any key to continue . . .
```

3. The system will display all the Patients that have made the appointment only.

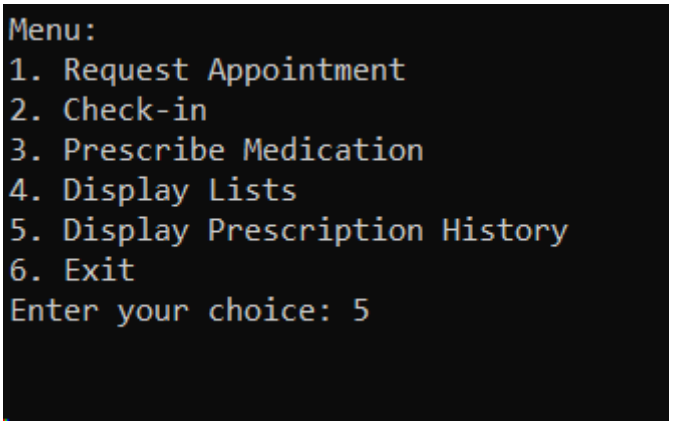
To export the history of the appointment hospital.

```
Menu:
1. Request Appointment
2. Check-in
3. Prescribe Medication
4. Display Lists
5. Display Prescription History
6. Exit
Enter your choice: 5
```

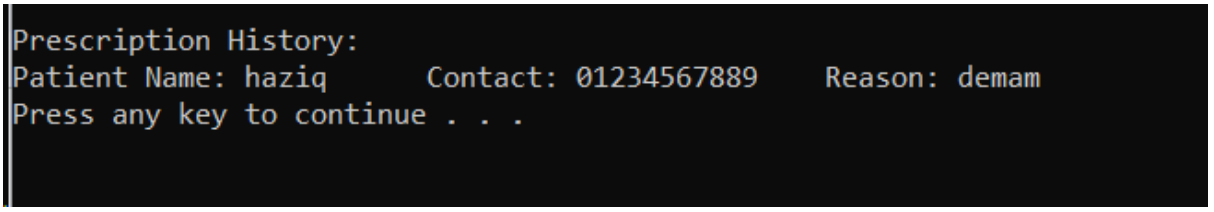
1. The system will display a menu.
2. Choose the (key in 5) option which will display all patients that have done an appointment.

3. The system will automatically export data into a text file.
4. You can find the txt file in the C directory in folder History.

To view the history of the Hospital appointment



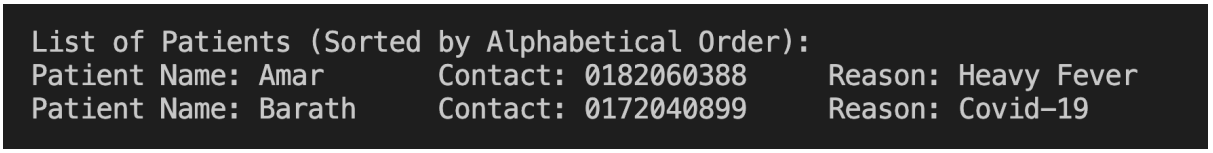
1. The system will display a menu.
2. Choose (key in 5) option 5 to view the list of all patients that do a discussion with the doctor.



3. The system will display all the Patients that have made the appointment only

To view history of the Patients (Alphabetical Order)

1. The system will display menu.
2. Choose (Key in 6) option 6 to view all patients history based on Alphabetical Order



3. The system will display all Patients that have made appointments.

To delete history entry of a Patient

```
Menu:
1. Request Appointment
2. Check-in
3. Prescribe Medication
4. Display Lists
5. Display Prescription History
6. Display Prescription History (Sorted by Alphabetical Order)
7. Delete Prescription History
8. Exit
Enter your choice:
```

1. The system will display menu.
2. Choose (Key in 7) option 7 to delete history entry of a Patient

```
Enter your choice: 7
Enter the name of the patient to delete: Amar
Patient Amar's history deleted successfully.
```

3. Input intended patient name in the space.
4. If patient name is found, success message will appear

```
Enter your choice: 7
Enter the name of the patient to delete: NoName
Patient NoName not found in the system.
```

5. If the patient name is not found, the program will notify the user that the patient name is not found in the system.

End the program

```
Menu:
1. Request Appointment
2. Check-in
3. Prescribe Medication
4. Display Lists
5. Display Prescription History
6. Exit
Enter your choice: 6
```

1. The system will display a menu.
2. Choose (key in 6) option 6 to exit the system.
3. The system will remove all the patient records that are still in the queue and the history.

```
Exiting program.
Press any key to continue . . .
```

4. The system will display a message before the end of the program