FACULTY OF COMPUTING

SEMESTER 1 2024/2025

MINI GROUP PROJECT

SECJ2013 - DATA STRUCTURE AND ALGORITHM

SECTION 02

LECTURER: DR ZURAINI BINTI ALI SHAH

NAME	**MATRIC NUMBER**
DHESHIEGHAN A/L SARAVANA MOORTHY	A23CS0072
LAU YAN KAI	A23CS0098
TEH RU QIAN	A23CS0191
NURUL ADRIANA BINTI KAMAL JEFRI	A23CS0258

Presentation Slides:
https://www.canva.com/design/DAGb9DJkbkc/CIP8oB9QSTYcBXf_nnYySA/view

Table of Content

1.0 Introduction

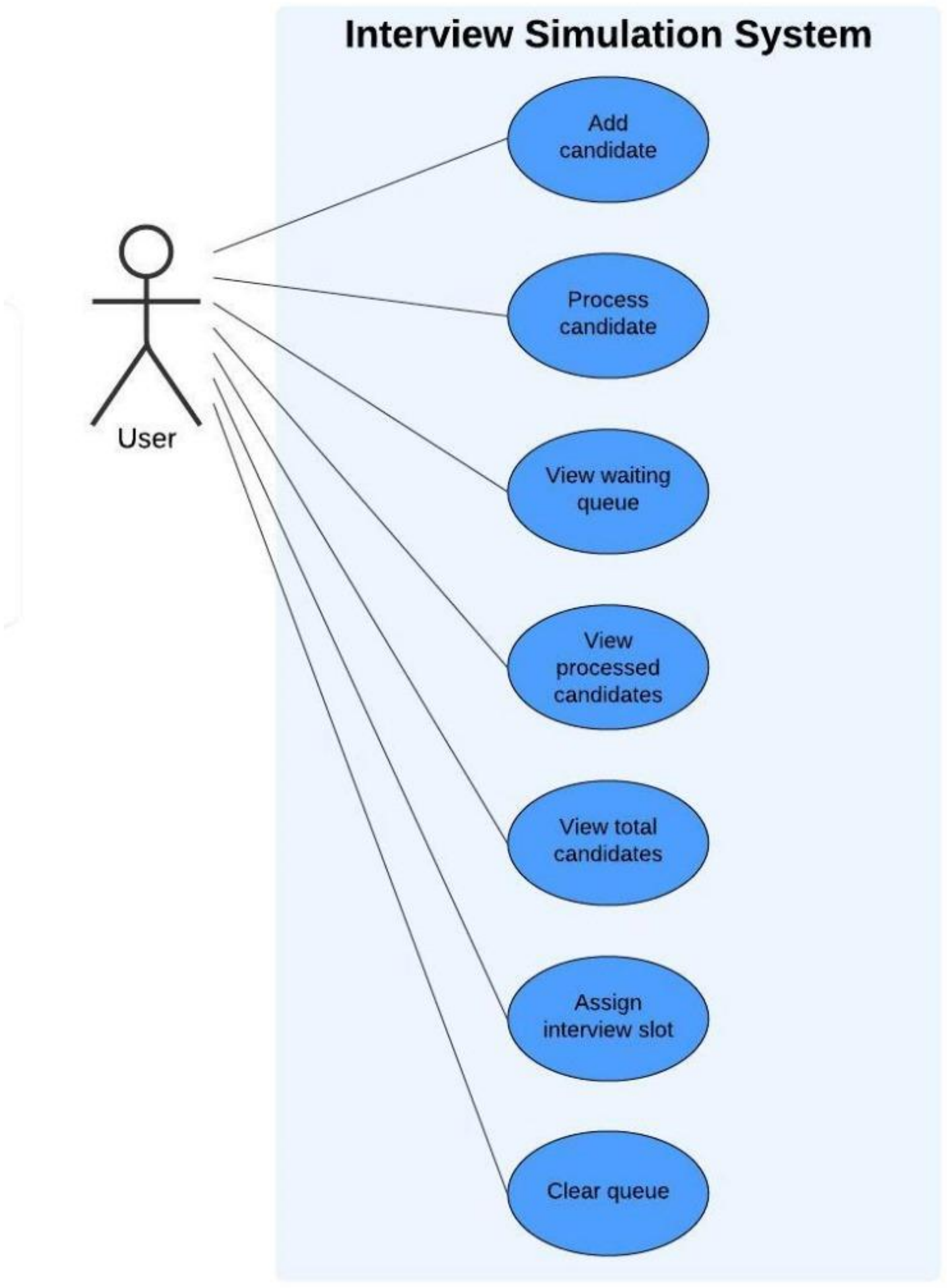
1.1 Synopsis

The Interview Simulation System was created to simplify and automate the interview scheduling process for candidates. It uses a queue-based data structure to keep track of applicants awaiting interview times. The system has a menu-driven interface for adding applicants, conducting interviews, assigning time slots, viewing current queue status and keeping track of the overall number of candidates in the queue.

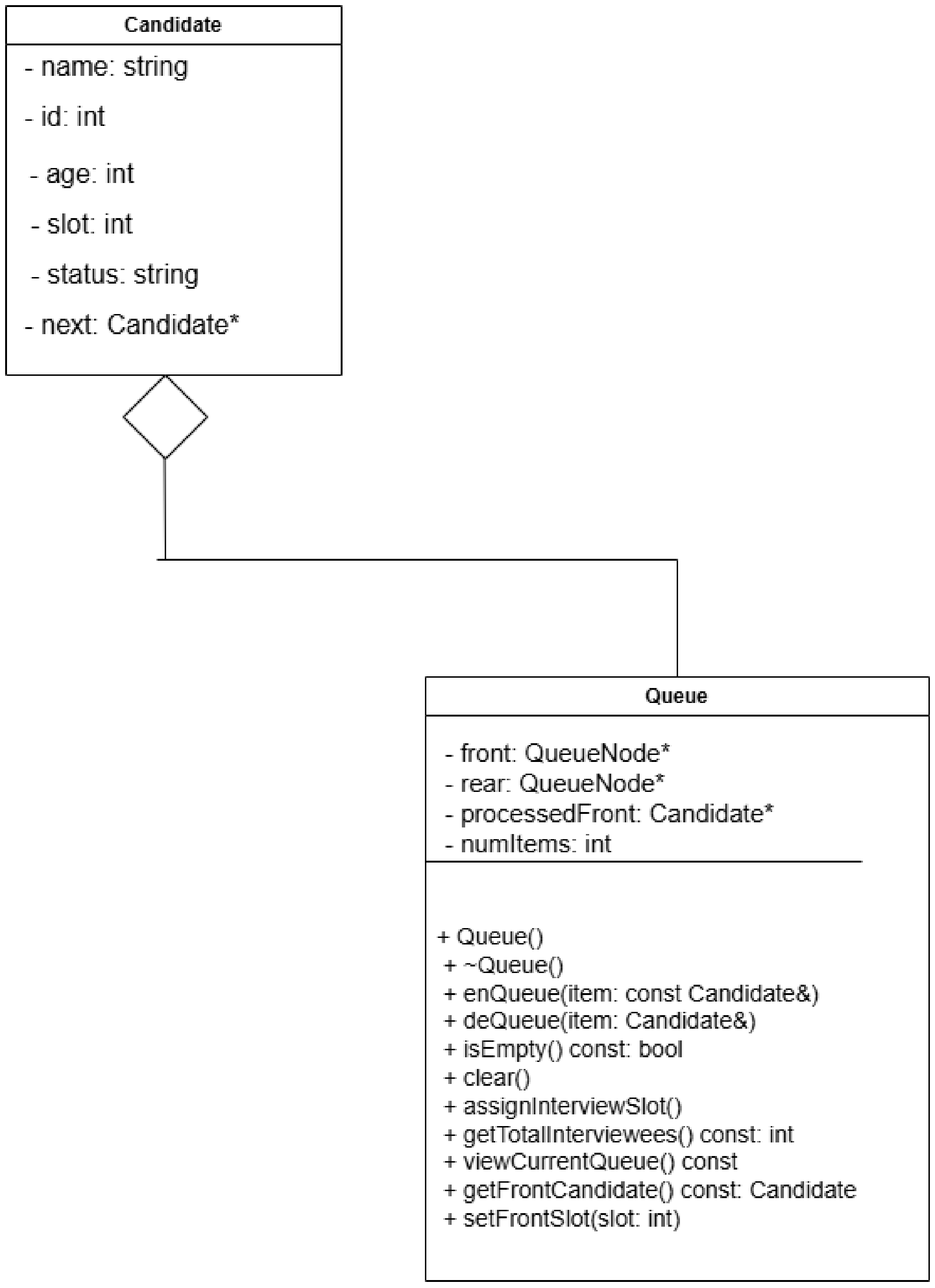
1.2 Objective

- To effectively manage candidates through the use of a queuing system.
- To assign interview dynamically to reduce the need for human scheduling.
- To have a clear distinction between "waiting" and "processed" candidates for improved visibility.
- To have a user-friendly interface that makes it simple to navigate the system's functions.

2.0 System Requirement



3.0 Analysis & Design (class diagram)



4.0 Development Activities and Coding in C++

4.1 Example Output of the System Interface

The following figures shows the completed interface of the Interview Simulation System.

```
=== Interview Simulation System ===
1. Add New Candidate
2. Process Next Candidate & Assign Slot
3. View Current Queue
4. View Total Candidates
5. Exit

Enter your choice (1-5): 1

Enter candidate name: Lee Chiu Sheng

Assigned ID: 1001

Enter candidate age: 21

Candidate added successfully!
Time Slot: Waiting to be processing

Processing application...
```

Screen 1: Add New Candidate – Candidate 1

Screen 1: The users select 1 to add new candidate. User needs to enter the candidate’s name, an ID will be assigned automatically, and the user is prompt to enter the candidate age. After entering all the values, candidate is added into the queue and waiting to be processing.

```
=== Interview Simulation System ===
1. Add New Candidate
2. Process Next Candidate & Assign Slot
3. View Current Queue
4. View Total Candidates
5. Exit

Enter your choice (1-5): 3
```

Screen 2: View Current Queue

Screen 2: After a new candidate is added, the user will be redirected back to the main menu, and the user can view the current interview queue by entering 3 from the main menu.

```
=== Interview Queue Status ===

WAITING CANDIDATES:
--------------------------------
Name: Lee Chiu Sheng
ID: 1001
Age: 21
Time Slot: Waiting to be processed
Status: Waiting
--------------------------------

Press Enter to return to main menu...
```

Screen 3: Current Interview Queue Status

Screen 3: The current interview queue will be displayed. One new added candidate is in the waiting list, waiting to be assigned in an interview slot.

```
=== Interview Simulation System ===
1. Add New Candidate
2. Process Next Candidate & Assign Slot
3. View Current Queue
4. View Total Candidates
5. Exit

Enter your choice (1-5): 1

Enter candidate name: Tan Yin Jia

Assigned ID: 1002

Enter candidate age: 24

Candidate added successfully!
Time Slot: Waiting to be processing

Processing application...
```

Screen 4: Add New Candidate – Candidate 2

Screen 4: The users may add a new candidate into the queue by selecting 1. User needs to enter the candidate's name, an ID will be assigned automatically, and the user is prompt to enter the candidate age. After entering all the values, candidate is added into the queue and waiting to be processing.

```
=== Interview Queue Status ===

WAITING CANDIDATES:
--------------------------------
Name: Lee Chiu Sheng
ID: 1001
Age: 21
Time Slot: Waiting to be processed
Status: Waiting
--------------------------------
Name: Tan Yin Jia
ID: 1002
Age: 24
Time Slot: Waiting to be processed
Status: Waiting
--------------------------------

Press Enter to return to main menu...
```

Screen 5: Current Interview Queue Status

Screen 5: The current interview queue will be displayed. One more new added candidate is in the waiting list. The new added candidate is assigned with different ID, and now waiting to be assigned in an interview slot.

```
=== Interview Simulation System ===
1. Add New Candidate
2. Process Next Candidate & Assign Slot
3. View Current Queue
4. View Total Candidates
5. Exit

Enter your choice (1-5): 4


Total candidates in queue: 2

Press Enter to return to main menu...
```

Screen 6: View Total Candidate

Screen 6: After added two candidates, when the user enters 4 to view the total candidates in the queue for interview sessions, it will show total of 2 are currently in the queue.

```
=== Interview Simulation System ===
1. Add New Candidate
2. Process Next Candidate & Assign Slot
3. View Current Queue
4. View Total Candidates
5. Exit

Enter your choice (1-5): 2
```

Screen 7: Process Candidate and Assign Interview Slot

Screen 7: After viewing total candidate in the queue, the user will be redirected back to the main menu, and the user can process the candidate by entering 2 from the main menu.

```
Processing next candidate:
--------------------------------
ID: 1001
Name: Lee Chiu Sheng
Age: 21
--------------------------------

Available Time Slots:
1. 9:00 AM - 10:00 AM
2. 10:30 AM - 11:30 AM
3. 1:00 PM - 2:00 PM
4. 2:30 PM - 3:30 PM
5. 4:00 PM - 5:00 PM

Select interview slot (1-5): 2

Candidate processed successfully!
Assigned Time: 10:30 AM - 11:30 AM

Processing...
```

Screen 8: Selecting Interview Time Slot for the Candidate

Screen 8: The first in the queue will be processed, and will be prompted to select the time slots available for interview sessions.

```
=== Interview Queue Status ===

WAITING CANDIDATES:
--------------------------------
Name: Tan Yin Jia
ID: 1002
Age: 24
Time Slot: Waiting to be processed
Status: Waiting
--------------------------------

PROCESSED CANDIDATES:
--------------------------------
Name: Lee Chiu Sheng
ID: 1001
Age: 21
Time Slot: 10:30 AM - 11:30 AM
Status: Processed
--------------------------------

Press Enter to return to main menu...
```

Screen 9: View Current Interview Queue Status

Screen 9: After a candidate is processed and assigned a time slot, it will be removed from the waiting list, and be moved to the processed candidate.

```
Exiting the Interview Simulation System...
Goodbye!

Press any key to continue . . .
```

Screen 10: Exit Programme

Screen 10: User may exit the programme after complete all the functions, by selecting 5 from the main menu.

4.2 Source Code

Available on: https://github.com/lauyankai/Interview-Simulation

```
#include<iostream>
#include <windows.h>
using namespace std;

struct Candidate {
    string name;
    int id;
    int age;
    int slot;
    string status;
    Candidate *next;
};

const string TIME_SLOTS[] = {
    "9:00 AM - 10:00 AM",   // Slot 1
    "10:30 AM - 11:30 AM", // Slot 2
    "1:00 PM - 2:00 PM",    // Slot 3
    "2:30 PM - 3:30 PM",    // Slot 4
    "4:00 PM - 5:00 PM"     // Slot 5
};

class Queue{
    private:
        Candidate* front;
        Candidate* rear;
        Candidate* processedFront;
        int numItems;

    public:
    //Constructor
    Queue() : front(nullptr), rear(nullptr), processedFront(nullptr),
numItems(0) {}

    //Destructor
    ~Queue() {
        clear();
        while (processedFront != nullptr) {
            Candidate* temp = processedFront;
            processedFront = processedFront->next;
            delete temp;
        }
    }

    //Destroy Queue
    void enQueue(const Candidate& item){
        Candidate* newNode = new Candidate;
        *newNode = item;
        newNode->next = nullptr;

        if(isEmpty()){
            front = newNode;
            rear = newNode;
        } else{
            rear->next = newNode;
            rear = newNode;
        }
        numItems++;
    }

    void deQueue(Candidate& item) {
        Candidate* temp = nullptr;

        if (isEmpty()) {
            cout << "The queue is empty.\n";
        } else {
            item = *front;

            Candidate* processedNode = new Candidate;
            *processedNode = item;
            processedNode->status = "Processed";
            processedNode->next = nullptr;

            if (processedFront == nullptr) {
                processedFront = processedNode;
            } else {
                Candidate* current = processedFront;
                while (current->next != nullptr) {
                    current = current->next;
                }
                current->next = processedNode;
            }

            temp = front;
            front = front->next;
            delete temp;
            numItems = numItems - 1;
            if(front == nullptr) {
                rear = nullptr;
            }
        }
    }

    bool isEmpty() const{
        return numItems == 0;
    }

    void clear(){
        Candidate candidate;
        while (!isEmpty()){
```

```cpp
            deQueue(candidate);
        }
    }

    void assignInterviewSlot() {
        if (isEmpty()) {
            cout << "No candidates in the queue to assign an interview slot.\n";
        } else {
            cout << "\nAssigning slot for the first candidate in queue:\n";
            cout << "--------------------------------\n";
            cout << "Candidate Name: " << front->name << "\n";
            cout << "Candidate ID: " << front->id << "\n";
            cout << "\nAvailable Time Slots:\n";
            for (int i = 0; i < 5; i++) {
                cout << i + 1 << ". " << TIME_SLOTS[i] << "\n";
            }

            int selectedSlot;
            do {
                cout << "\nSelect interview slot (1-5): ";
                cin >> selectedSlot;
                if (selectedSlot < 1 || selectedSlot > 5) {
                    cout << "Invalid slot! Please select between 1-5.\n";
                }
            } while (selectedSlot < 1 || selectedSlot > 5);

            front->slot = selectedSlot;
            cout << "\nSlot assigned successfully!\n";
            cout << "Candidate: " << front->name << "\n";
            cout << "Assigned Time: " << TIME_SLOTS[selectedSlot - 1] << "\n";
            cout << "\nProcessing...";
            Sleep(3000);
            system("cls");
        }
    }

    int getTotalInterviewees() const {
        return numItems;
    }

    void viewCurrentQueue() const {
        if (!isEmpty()) {
            cout << "\nWAITING CANDIDATES:\n";
            cout << "--------------------------------\n";
            Candidate* current = front;
            while (current != nullptr) {
                cout << "Name: " << current->name << "\n";
                cout << "ID: " << current->id << "\n";
                cout << "Age: " << current->age << "\n";
                cout << "Time Slot: Waiting to be processed" << endl;
                cout << "Status: Waiting\n";
                cout << "--------------------------------\n";
                current = current->next;
            }
        }

        if (processedFront != nullptr) {
            cout << "\nPROCESSED CANDIDATES:\n";
            cout << "--------------------------------\n";
            Candidate* current = processedFront;
            while (current != nullptr) {
                cout << "Name: " << current->name << "\n";
                cout << "ID: " << current->id << "\n";
                cout << "Age: " << current->age << "\n";
                cout << "Time Slot: ";
                if (current->slot > 0 && current->slot <= 5) {
                    cout << TIME_SLOTS[current->slot - 1];
                } else {
                    cout << "Waiting to be processed";
                }
                cout << endl;
                cout << "Status: " << current->status << "\n";
                cout << "--------------------------------\n";
                current = current->next;
            }
        }

        if (isEmpty() && processedFront == nullptr) {
            cout << "\nNo candidates in the system.\n";
        }
    }

    Candidate getFrontCandidate() const {
        if (!isEmpty()) {
            return *front;
        }
        throw runtime_error("Queue is empty");
    }

    void setFrontSlot(int slot) {
        if (!isEmpty() && slot >= 1 && slot <= 5) {
            front->slot = slot;
        }
    }
};

int main() {
    Queue interviewQueue;
    int choice;
    int nextId = 1001;

```

```cpp
    do {
        system("cls");
        cout << "\n=== Interview Simulation System ===\n";
        cout << "1. Add New Candidate\n";
        cout << "2. Process Next Candidate & Assign Slot\n";
        cout << "3. View Current Queue\n";
        cout << "4. View Total Candidates\n";
        cout << "5. Exit\n";
        cout << "\nEnter your choice (1-5): ";
        cin >> choice;

        switch (choice) {
            case 1: {
                Candidate newCandidate;
                cin.ignore();

                cout << "\nEnter candidate name: ";
                getline(cin, newCandidate.name);

                newCandidate.id = nextId++;
                cout << "\nAssigned ID: " << newCandidate.id << endl;

                cout << "\nEnter candidate age: ";
                cin >> newCandidate.age;

                newCandidate.status = "Waiting";
                newCandidate.next = nullptr;

                interviewQueue.enQueue(newCandidate);
                cout << "\nCandidate added successfully!\n";
                cout << "Time Slot: Waiting to be processing\n";
                cout << "\nProcessing application...";
                Sleep(3000);
                system("cls");
                break;
            }

            case 2: {
                Candidate processedCandidate;
                if (!interviewQueue.isEmpty()) {
                    try {
                        Candidate frontCandidate =
interviewQueue.getFrontCandidate();
                        cout << "\nProcessing next candidate:\n";
                        cout << "--------------------------------\n";
                        cout << "ID: " << frontCandidate.id << endl;
                        cout << "Name: " << frontCandidate.name << endl;
                        cout << "Age: " << frontCandidate.age << endl;
                        cout << "--------------------------------\n";

                        cout << "\nAvailable Time Slots:\n";
                        for (int i = 0; i < 5; i++) {
                            cout << i + 1 << ". " << TIME_SLOTS[i] <<
"\n";
                        }

                        int selectedSlot;
                        do {
                            cout << "\nSelect interview slot (1-5): ";
                            cin >> selectedSlot;
                            if (selectedSlot < 1 || selectedSlot > 5) {
                                cout << "Invalid slot! Please select
between 1-5.\n";
                            }
                        } while (selectedSlot < 1 || selectedSlot > 5);

                        interviewQueue.setFrontSlot(selectedSlot);

                        interviewQueue.deQueue(processedCandidate);

                        cout << "\nCandidate processed successfully!\n";
                        cout << "Assigned Time: " <<
TIME_SLOTS[selectedSlot - 1] << "\n";
                        cout << "\nProcessing...";
                        Sleep(3000);
                    } catch (runtime_error& e) {
                        cout << e.what() << endl;
                        Sleep(2000);
                    }
                } else {
                    cout << "\nNo candidates in the queue to process.\n";
                    Sleep(2000);
                }
                system("cls");
                break;
            }

            case 3: {
                system("cls");
                cout << "\n=== Interview Queue Status ===\n";
                interviewQueue.viewCurrentQueue();
                cout << "\nPress Enter to return to main menu...";
                cin.ignore();
                cin.get();
                system("cls");
                break;
            }

            case 4: {
                system("cls");
                cout << "Total candidates in queue: "
                     << interviewQueue.getTotalInterviewees() << endl;
                cout << "\nPress Enter to return to main menu...";
                cin.ignore();
```

```cpp
                cin.get();
                system("cls");
                break;
            }

            case 5: {
                system("cls");
                cout << "\n\nExiting the Interview Simulation System... Goodbye!\n\n";
                break;
            }

            default: {
                cout << "Invalid choice! Please try again.\n";
                system("cls");
                break;
            }
        }
    } while (choice != 5);

    return 0;
}
```

5.0 Task distribution

Task	Person in Charge
Implement queue operations (Code)	Ru Qian
Create candidate class (Code)	Adriana
Implement slot assignment (Code)	Dhesh
Integrate all and implement the main menu interface (Code)	Yan Kai
Synopsis and objective	Ru Qian
System requirement	Dhesh
Analysis & design	Adriana
Development activities and coding in C++	Yan Kai

--- END OF DOCUMENTATION ---