CSE108 – Computer Programming Lab.
Lab 12
Selection Operations
Due at 10am.

Hand in: A student with number 20180000001 should hand in a zip file named 20180000001.zip for this lab.

Overview

This assignment is about the implementation of a **priority queue** using a **doubly linked list**. You will design a queue system that simulates a bank customer system. The functionalities required for the system involve creating, reading, updating, and deleting data related to the customers.

Part 1.   (60 points)

In the first part of the assignment, you will implement the basic operations of the queue. This includes implementing a **priority queue** and making sure the customer data can be read from and written to a text file.

Customer Data Structure (15 points)

You will create a struct named **Customer** in C, which contains the following properties:

Name (string)

Age (integer)

Priority Group Name (string)

Priority Level (integer)

Default Priority Levels are following:

- *VIP Customers:* Priority Level **4** (Represented by '**VIP**')
- Old or Disabled Customers: Priority Level **3** (Represented by '**OD**')
- Veteran Customers: Priority Level **2** (Represented by '**VETERAN**')
- Normal Customers: Priority Level **1** (Represented by '**NORMAL**')

Order of the customers that are in the same group should be established by considering **FIFO(First In First Out)** rule.

File Operations (15 points)

All customer information should be kept in a text file named **customers.txt**. Write functions to read from this file and insert the data into your priority queue. The priority queue should be constructed such that the customer with the highest priority level is at the front.

Priority Queue Operations (30 points)

Write functions to perform the following operations on the priority queue:

Insert new customer (10 points): Implement an **insertCustomer** function that inserts a new customer into the queue according to their priority level. After insertion, the data in the queue should be written back into the customers.txt file.

Remove Customer (10 points): Implement a **removeCustomer** function that deletes a customer from the queue. In order to do so, enter the name of the customer to remove. After removal, the data in the queue should be updated in the customers.txt file.

Display customers in order (10 points): Implement a **displayCustomers** function that prints the customers in the queue with respect to their priorities.

Part 2.   (40 pts.) In the second part of the assignment, you will implement additional features for your queue.

Searching Customers (15 points)

Implement a recursive function **searchCustomer** that allows users to search for a customer by name. This function should display all the details of the customer if found.

Priority Settings (25 points)

The priority level of each customer should be modifiable during runtime. For exceptional cases, your program should be able to change a specific customer's priority level. To do so, the function should find the customer by using his or her name. Implement a function named **changePriority** for this part.

`Example Outputs:`

```
**************************************************
Welcome to the ABC Bank

1-Insert new customer
2-Remove customer
3-Display customers in order
4-Search a customer
5-Change priority settings
6-Exit

Please select an option: 1
…
Please enter customer's Name: John Doe
Please enter customer's Age: 35
Please enter customer's Priority Group: NORMAL
Customer added successfully…
```

```
**************************************************
Welcome to the ABC Bank

1-Insert new customer
2-Remove customer
3-Display customers in order
4-Search a customer
5-Change priority settings
6-Exit

Please select an option: 2
…
Please enter customer's Name: John Doe
Customer removed successfully…
```

```
************************************************
```
Welcome to the ABC Bank

1-Insert new customer
2-Remove customer
3-Display customers in order
4-Search a customer
5-Change priority settings
6-Exit

Please select an option: 3
…
Customers:
1. Jane Doe - Age: 45 - Priority Group: VIP - Priority Level: 4
2. John Smith - Age: 70 - Priority Group: OD - Priority Level: 3
3. Bob Johnson - Age: 30 - Priority Group: NORMAL - Priority Level: 1

```
************************************************
```
Welcome to the ABC Bank

1-Insert new customer
2-Remove customer
3-Display customers in order
4-Search a customer
5-Change priority settings
6-Exit

Please select an option: 4
…
Please enter customer's Name: John Smith
Found: John Smith - Age: 70 - Priority Group: OD - Priority Level: 3
…

```
****************************************************
Welcome to the ABC Bank

1-Insert new customer
2-Remove customer
3-Display customers in order
4-Search a customer
5-Change priority settings
6-Exit

Please select an option: 5
…
Please enter customer's Name: Bob Johnson
Please enter new Priority Level: 3
Priority level changed successfully…
New properties are following:
Bob Johnson - Age: 30 - Priority Group: Normal - Priority Level: 3
…
```

General Rules:

1. You will have two hours to provide a solution to the given problem set.
2. You will be able to hand in your solutions via Teams in the next two hours. The submission will be closed exactly at 10am.
3. There will be an interview session immediately after the submission deadline. Starting at 10am, you will be randomly invited to attend a meeting by a TA to demonstrate your solution and answer any questions asked by the TA.
4. You must be available until 1pm to respond to the demo invitation whenever you receive it. You will have 3 minutes after you are called via Teams. If you do not answer/appear in 3 minutes, you will miss you interview.
5. If you miss your interview or are unable to give satisfactory answers to the questions, you will receive a zero for that lab even if you have submitted your solution.
6. If you have not submitted a solution in time, you will not be invited for the interview and receive zero for that lab.
7. Due to time constraints, some students may not be invited to an interview. In that case, their solutions will be graded offline.
8. Unless you aren't declared for a specific prototype, you may use arbitrary but proper function and variable names that evoke its functionality.
9. The solution must be developed on given version of OS and must be compiled with GCC compiler, any problem which rises due to using another OS or compiler won't be tolerated.

10. Note that if any part of your program is not working as expected, then you can get zero from the related part, even it is working partially.
11. Zip your solution file before uploading it to MS Teams. The zip file must contain the C file with your solution and screenshots of the valid outputs of the program.