VIETNAM NATIONAL UNIVERSITY - HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
**FACULTY OF COMPUTER SCIENCE AND ENGINEERING**


## ASSIGNMENT 1

# JJK RESTAURANT OPERATIONS

## (PART 1)

HO CHI MINH CITY, SEPTEMBER 2023

# Assignment 1

**Version 1.0**

## 1 Outcomes

After finishing the assignment, students can:

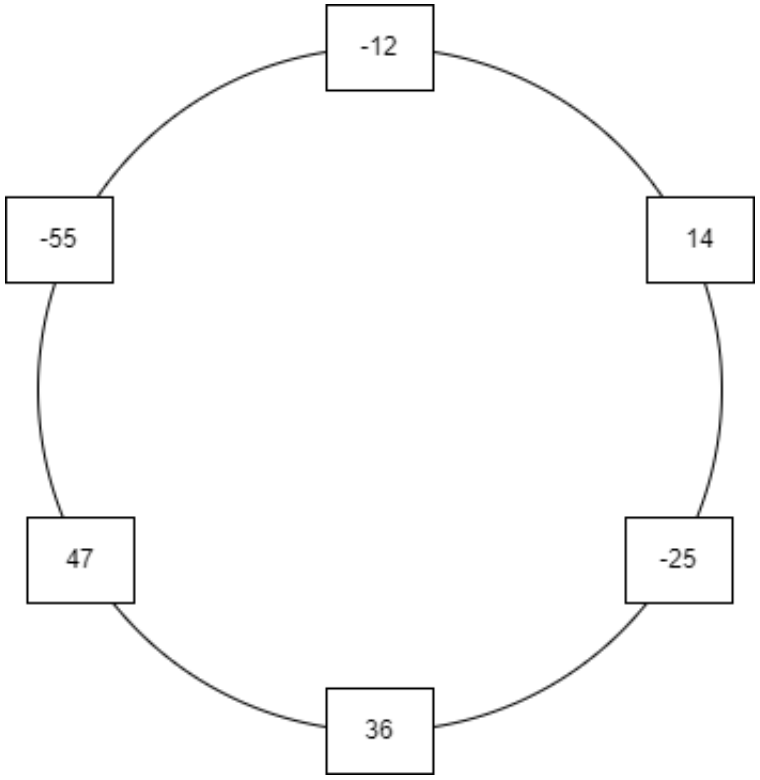
- Implement linked list structures.
- Choose and apply the appropriate data structures to achieve the desired results.
- Know about a famous manga.

## 2 Introduction

In this assignment, students will simulate the process of handling bookings and arranging seating for Sorcerers and Curses (referred to as "customers") in a restaurant. This simulation will be carried out using commands as described in Section 2.1. The restaurant is owned by Gojo and Sukuna.


Hình 1: Image at restaurant.

**Note**: The restaurant do not use the number to determine seating for customers, but will arrange it flexibly based on customer information.Moreover, after processing all the commands in the input file and outputting the results to the screen, the program must destroy all dynamically allocated data objects, ensuring no garbage in memory before ending program.




Hình 2: Circular doubly linked list is used to represent the seats in restaurants, basing on ENERGY.

The meanings of the abbreviations and data types of the parameters are described as follows:

- **NAME**: a continuous string of Alphabet characters (including lowercase and uppercase letters) without spaces, representing the customer's name.
- **ENERGY**: an integer, representing the energy of Sorcerers (positive value) and Curses (negative value).
- **NUM**: an integer with a different meaning and range of value for each different command.
- **MAXSIZE**: a maximum number of seating that the restaurant can serve.

## 2.1 List of commands

**RED <NAME> <ENERGY>:**

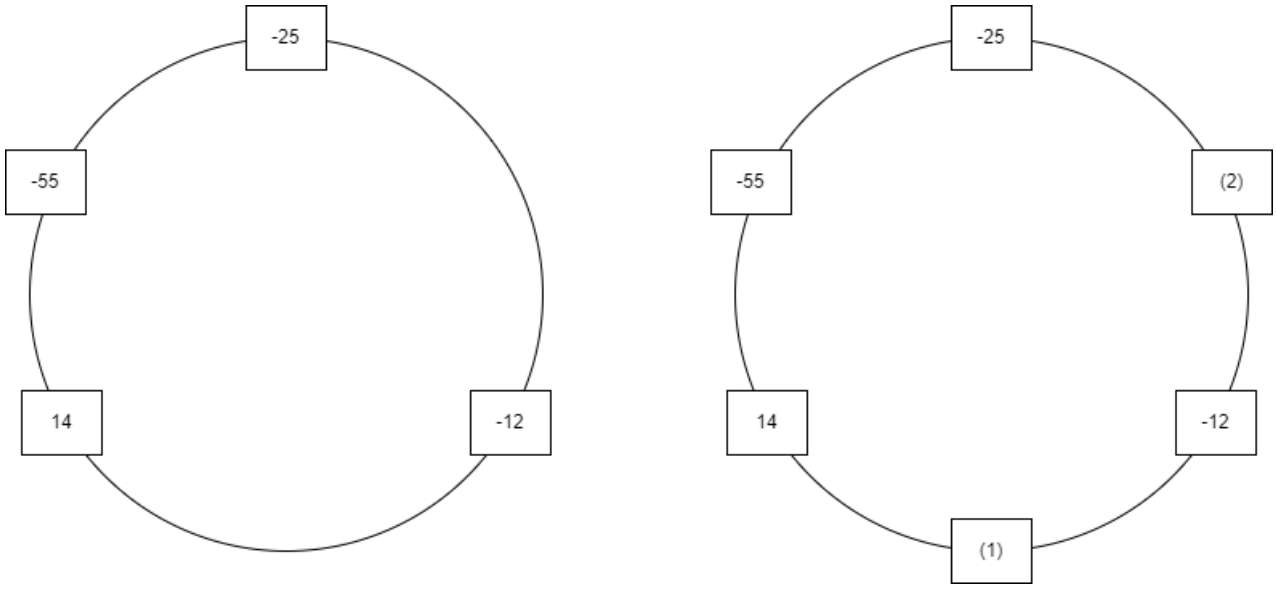
Hình 3: Cursed Technique: RED. [4]

This command is utilized by staff to record customer information, thereby arranging seating positions for customers.

Customers coming to the restaurant have to provide information to the staff so the staff can arrange seating. In some cases, staff will arrange seats based on the customer's ENERGY value. However, the restaurant also has some regulations regarding seating arrangements as follows:

- The restaurants only accepts customer, who are Sorcerers (positive ENERGY) or Curses (negative ENERGY). When the restaurant opens, the first customer may choose any seat at the round table. After that, the seating arrangement for the new guest will be **calculated from the nearest position that has just changed** (sitting or leaving). Assume it is called position X.

- When the next guest arrives, the staff will arrange a seat for the guest at an **adjacent location on the clockwise side** if the guest's ENERGY is greater than or equal to the guest's ENERGY at position X. Otherwise, the guest will sit at an adjacent location on the counterclockwise side of the customer at position X.
- For example, suppose that the customer at position X has ENERGY = -12 (left image) and is being considered for adding an element. Position (1) is considered the position adjacent to element X on the clockwise side, position (2) is considered the position adjacent to element X on the counterclockwise side (right image).




Hình 4: Demonstrate the adding adjacent customer around X's position.

- However, when the number of guests at the table is greater than or equal to MAXSIZE/2, staff will change their strategy for choosing seats. Because they know that customer with nearly equal ENERGY often do not like to sit close to each other. Therefore, before choosing a seat, staff will calculate the largest difference between the new customer and all the customers in the restaurant by taking the absolute value of each ENERGY pair for each customer (assume, called **RES**) to determine seating.
  - If multiple positions yield the same RES value after calculation, the first clockwise position will be selected. If only one position has the largest RES value, choose that position.
  - After that, staff will remove the absolute value sign of RES. If the result is negative, add it to the adjacent counterclockwise side; otherwise, add it to the adjacent clockwise side at the found position.
- Furthermore, because of "In heaven, in the world, I am the one and only", the restaurant

does not accept latecomers with names similar to those already dining or in the queue. For instance, if a customer named **ABC** is currently dining or in the queue, subsequent customers with the same name, **ABC**, will be reject to enter the restaurant.

- When the number of customers at the table reaches MAXSIZE, the restaurant will halt further admissions and place customers in a queue. The maximum queue capacity is an integer, also set to MAXSIZE. When the queue reaches its maximum capacity, the restaurant will stop receiving additional customers.

**BLUE <NUM>**:

Hình 5: Cursed Technique: BLUE. [4]

This command is used for staff to remove customers who have overstayed and to clean the table. After cleaning, if there are customers in the queue, staff will allocate seats to them; otherwise, no further action is taken

When receiving the command, the staff will remove NUM guests in the order of entering the restaurant from earliest to most recent. For example, if customers come to the restaurant in the order $A \longrightarrow B \longrightarrow C \longrightarrow D$, after executing command **BLUE 2**, there are only two customers left in the restaurant, C and D. If NUM is greater than or equal to the number of guests at the table or greater than MAXSIZE, it is considered that the restaurant owner decides to remove all guests from the table.

Please note that allocating a new seat for a customer in the queue can only occur after the BLUE command has been executed. The seat selection mechanism for guests is similar to the RED command, following a First In First Out (FIFO) order from the queue. Additionally, it is guaranteed that the value of NUM will always be greater than 0.

**PURPLE**:

Hình 6: Cursed Technique: PURPLE.[2]

PURPLE is used to re-arrange customers in the queue, following the descending order of the absolute value of ENERGY (starting from the beginning of the queue) using the Shell Sort algorithm. To implement this, please refer to the Shell Sort algorithm as described in the reference book [5], chapter 7.3

When receiving this order, the staff will proceed as follows:

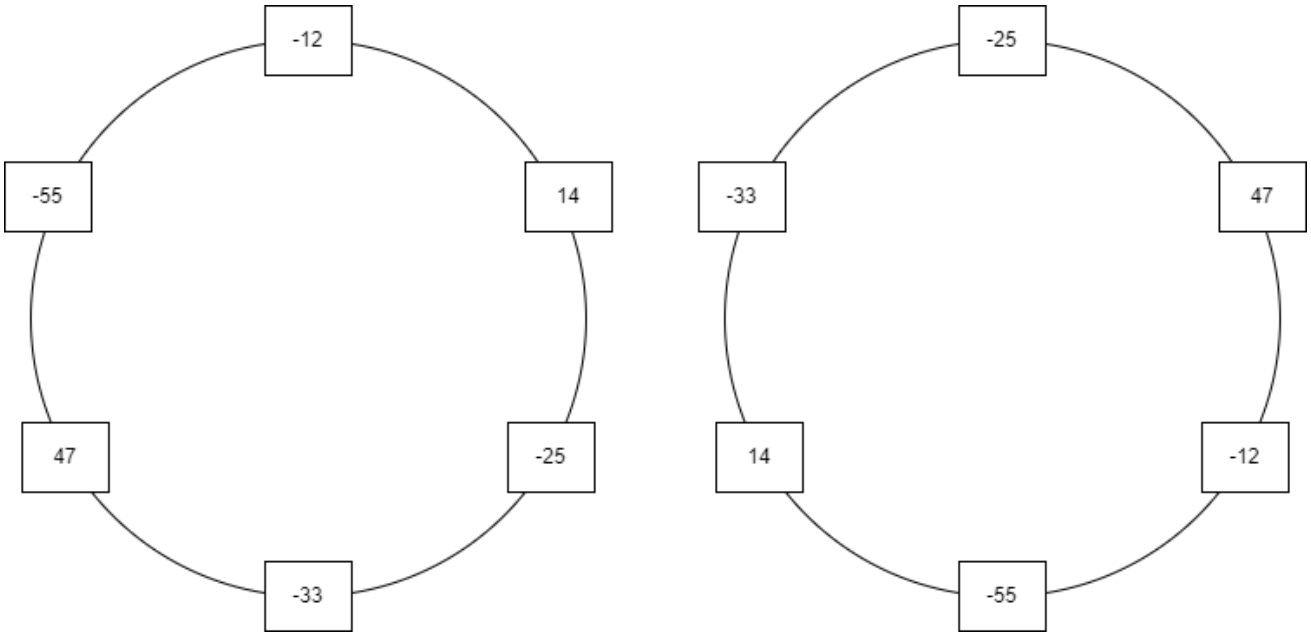
- First, find the position of the customer with the highest absolute value of ENERGY in the queue. If there are multiple customers with the same highest absolute value of ENERGY, choose the guest most recently added to the queue.
- Then, re-arrange the queue **ONLY** from the found position to the beginning of the queue. **In case customers have the same ENERGY, the customer who most recently adđe to the queue is considered larger.**
- Assume the number of position transitions in the queue when performing sorting using the Shell Sort algorithm as N. The staff will continue to execute the command **BLUE <N mod MAXSIZE>**

**REVERSAL**:

Starting from position X and considering the counterclockwise direction, reverse the positions of customers in the restaurant.

However, staff can only reverse the positions of Sorcerers or Curses and cannot interchange the positions between them.

The image on the left side illustrates an example of customer positions. Assuming that position X corresponds to ENERGY = -12, executing the REVERSAL command will result in the configuration shown on the right.




Hình 7: The result after executing REVERSAL (from left to right).

**UNLIMITED_VOID**:

Starting from position X and considering the clockwise direction, find the longest subsequence where the number of consecutive values must be **greater than or equal to 4** and the **total ENERGY value is the smallest** for the customers at the dining table. Print the guest information within that subsequence in the format "NAME-ENERGY/n", in clockwise direction starting from the smallest element within the subsequence. If multiple subsequences have the same value, choose the last one found in clockwise direction. If there is no subsequence meeting the requirements, no action is required. **DOMAIN_EXPANSION**:

Hình 8: UNLIMITED_VOID.[3]

Due to conflicts between Sorcerers and Curses resulting in a huge fight in the restaurant, the restaurant owner will decide to remove all Sorcerers or Curses if the following conditions are met:

- If the sum of ENERGY for all Sorcerers at the restaurant("customers are at the restaurant" means customers are at the table and in the queue) is greater than or equal to the absolute sum of ENERGY for all Curses, the staff will remove all the Curses from the restaurant. Otherwise, the staff will remove all the Sorcerers.
- Subsequently, if this results in empty seats, the staff will proceed to arrange customers from the queue, similar to the RED command, following the FIFO rule.
- Then, print the information of the removed Sorcerers or Curses in order from the most recent customer to the earliest customer, following the format: "NAME -ENERGY/n".

**LIGHT <NUM>:**

Print the information of all guests at the dining table, following clockwise direction and starting from position X if the NUM value is positive, or counterclockwise direction if the NUM value is negative. Use the format "NAME-ENERGY/n". If NUM is 0, print the information of customers in the queue in the order from the beginning to the end of the queue, also following the "NAME-ENERGY/n".

## 2.2 Conclusion

Due to a disagreement between the two restaurant co-owners, Gojo and Sukuna, concerning the best interests of our customers, the restaurant is temporarily closed for resolution. The outcome of this resolution will be addressed in the next major exercise. Hopefully nothing will have to be split in half.


Hình 9: An argument between the owners to protect their customers. [1]

### 2.3 Requirements

To finish the assignment, students need to:

- Download and unzip the initial.zip.
- There are 4 files: main.cpp, main.h, restaurant.cpp and test.txt.. You MUST NOT modify files main.cpp and main.h.
- Modify files Cache.h and Cahe.cpp to implement the cache memory.
- Make sure that there is only one **include** directive in restaurant.cpp that is **#include "main.h"**. No more include directive is allowed in this file.
- The assignment is used to test the use of linked lists, so students are not allowed to use any contents related to static arrays or dynamic arrays such as: *(arr), arr[i], ... even in comment. In addition, students **must use the print() function specified in the customer class** to print customer information to the screen. Because when grading, this print() function can be changed to check some other problems in the assignment.
- The environment used for grading is g++ (MinGW-W64 i686-ucrt-posix-dwarf) 11.2.0.

## 3 Submission

Students submit only 1 file: **restaurant.cpp**, before the deadline specified in the link "Assignment 1 Submission". There is simple test case used to check your solution to make sure your solution can be compiled and run. You can submit as many as you like but just the last submission is marked. As the system cannot response too many submissions in the same time, you should submit as soon as possible. You will take your own risk if you submit on the time of deadline. After the deadline, you cannot submit anymore.

## 4 Harmony

There is a question in the exam that are related to the content of the assignment. It will be clearly stated that the question is used to harmonize the assignment. The scores of harmony questions will be scaled to 10 and will be used to recalculate scores for the assignments. Specifically:

- Let x be the score of assignment.
- Let y be the score of harmony question after scaling to 10.

The final assignment score will be:

$$\textbf{Assignment\_Score} = \textbf{2 * x * y / (x + y)}$$

# 5 Other regulations

- Students must complete this assignment on their own and must prevent others from stealing their results. Otherwise, the student treat as cheating according to the regulations for cheating.
- Any decision made by the teachers in charge of this assignment is the final decision.
- Students are not provided with test cases after grading, students will be provided with the assignment's score distribution.

**THERE ARE NO EXPLANATIONS OR EXCEPTIONS.**

# References

[1] https://www.deviantart.com/

[2] https://www.pxfuel.com/

[3] https://www.wallpaperflare.com/

[4] https://wallpaperaccess.com/

[5] Data Structures & Algorithm Analysis by Clifford A. Shaffer - Edition 3.2 (C++ Version)

**——————————END——————————**