Order lists

VIP orders:

They were chosen to be in a priority queue so that it can be easily arranged by the priority equation.

The advantages of this choice are that we can insert in O(log(n)) we can also extract the max priority in O(1).

Frozen orders:

They were chosen to be in a normal queue so that we can get the order to serve which is always at the beginning of the queue.

The advantages of this choice are that we can insert in O(1) we can also delete the order we want(always at the beginning) in O(1).

Normal orders:

They were chosen to be in a linkedlist so that we can get the first element to be served, we can also search the list for an element to delete.

The advantages of this choice are that we can insert in O(1) if we have tail in the linked list implementation we can also delete in O(1) from the beginning we can search the list in O(n).

Motorcycles

All the different types of motorcycles were put in priority queues because it should be sorted by its speed and we will make the bonus of different speeds.

Each type of motorcycle has 2 priority queues one for the idle motors and the other for the serving motorcycles because each one has different priority the first priority is the speed the second is when will it arrive at the restaurant again.

The advantages of this choice are that we can insert in O(log(n)) we can also extract the max priority in O(1).

We have two additional lists, one for the events queue which are stored in a queue as they are ordered by the arrival time and they are FIFO, other list is a priority queue used for the orders being served so we can order them by with priority of the finish time once the finish time is reached for an order it can be written in the output file then deleted insert in O(log(n)) delete max priority in O(1).

MODES

INTERACTIVE-MODE&STEP-BY-STEP:

We took the Simulation Function that we have done before (Phase I) , and we modified if by functions which control the Assignment of Motorcycles ,we found that the **INTERACTIVE** and **STEP BY STEP** MODES are very similar so we made them in one function but we controlled them by an enumeration so we control if we need to wait for a click or wait a second .

SILENT MODE:

This mode will just print in the output file but the program will work in the background we modified a class output that print all the information about Restaurant (Orders, Statistics, Motorcycles).

Performance

GENGARL PROBLEMS:

First after we choose mode of the 3 modes we found that we must take inputs from a file so we decided to organize a class Input to take the input from file input and distribute the data on the data structure in **Each Region** , after thinking we found that we must make a class Region to make it easy to distribute the data and know the orders and motorcycles region in the runtime.

MODES PROBLEMS:

1. **Problem in interaction and step-by-step modes:**
   We found that we need to print the orders in each time step on the interface of the program so we found that the GUI has an Array which takes the orders and prints it on the interface so we filled that array with orders but First we filled VIP Orders ,Because it has the highest priority then Frozen, and finally Normal orders the filling of the array with a function called **LOADGUI( )** that we called it each time step to update the Array in GUI .

2. **Assignment problem in the 3 modes:**
   In each time step we must assign orders to motorcycles so we made function to make this job for us called **AssignMotorcycles( )** this function is called every time step to Assign the active orders to a motorcycles if exist and but the motorcycles in the served list and this list have motorcycles orders with it's back time in this function we Assign Motorcycles to orders in **all regions** we first assign VIP orders and then Frozen and Normal

3. **Un Assignment problem in the 3 modes:**
   In each time step we must check if the motorcycle that has been assigned an order reach the destination or not this to check if the motorcycle is back so we use it to serve another order and put motorcycles in its original list.

4. **Printing:**
   In each time step after we unassign motorcycles we found that we can print in Output file the orders which reached its finish time and delete it so we don't waste memory to the end of program .
   We also modified the print message function in GUI to support multi-lines by taking arguments in its parameter list and print those arguments in multi-lines .

5. **Cancellation & Promotion events :**

We found that we need to loop the Normal active orders list to find the order which need to be promoted or canceled so we added for the data structures this function **TOARRAY( )** to make the cancellation and promotion in O(n) and we used it to in the filling of the GUI Array.

6. **Counters:**

We needed to count the active orders in each time step and we need to count the whole orders in each region so we modified the class region by counter to the active order and whole orders to print them in each time step .

7. **Motorcycles Damage and Traffic Problems :**
   We made a list for tired motorcycles and in each time step we check if they recover by function recover and in Un Assign function we check after he back if he is tired we insert him in the tired list so he waits for recovering.