Data Structures and Algorithms
Final Assessment Report

Team Name: Team 12 | **Number of members: 4**

Email:

Section1: Andrew Boshra, Sec:1, B.N:17

- Restaurant::Modes()

Member of: Class Restaurant

Inputs:

mode :PROG_MODE enum represents the program's mode

Returns: nothing

Called By:

- Restaurant::RunSimulation( )
-

Calls:

- Restaurant::ExecuteEvents(CurrentTimeStep)
- Restaurant::ExecuteEvents(CurrentTimeStep)
- Restaurant:: Cook_Statues(CurrentTimeStep)
- Restaurant:: FreeOrder(CurrentTimeStep,int&NOrders,int&VOrders,int&GOrders)
- Restaurant:: AssignOrder(CurrentTimeStep,LinkedList<Cook*>LastAssigned)
- Restaurant:: Cook_Injury(CurrentTimeStep)
- Restaurant:: FillDrawingList()
- GUI::printMessage()
- GUI:: ResetDrawingList()

Function Logic description:

- This function controls simulation in all modes
- Loops untill all orders are finished and all events are excuted and increases timestep in each loop.
- In each timestep it :
    - Excute events in this time step.
    - Assign orders to available cooks.
    - Changes cook status and it may make cook status injured.
    - Changes finished orders status to DONE.
    - Updates the interface if it is in interactive or stepbystep mode.
    - Waits for a click in interactive modes.
    - Waits for 1 second in Step by Step mode.
    - Move to the next time step.

-Restaurant :: CreateOutputFile()

Member of: Class Restaurant
Inputs: nothing
Returns: nothing

Called By:
- Restaurant::RunSimulation( )

Calls:
- selectionSort()
- Order:: getWaitingTime()
- Order:: getServingTime()
- Order:: is_urgent()

Function Logic description:
- This functions Creates Output file called "Output.txt" with the required format.
- It Creates an array of Order pointers and sorts them with finishing time.
- It also calculates the average serve time and average waiting time.
- It calculates number of auto promoted and urgent orders

- selectionSort(Order* arr[], int n)

Member of: Global function
Inputs:
- arr: Unsorted array of pointer to orders
- n: Number of Orders pointers Orders

Returns: nothing

Called By:
- Restaurant::CreateOutputFile()

Calls:
- Order:: getServingTime()
- Order::getFinishedTime()

Function Logic description:
- it sorts array of pointer to orders by finishing time
- if 2 orders finished at the same timestep they will be sorted by serving time

Section1: Ahmed khaled mahmoud, Sec:1, B.N:5

- Restaurant::FreeOrder (int CurrentTimeStep,int&NOrders,int&VOrders,int&GOrders)

Member of: Class Restaurant
Inputs:
CuurentTimeStep
NOrders : Number of finished normal orders
VOrders : Number of finished VIP orders
GOrders : Number of finished vegan orders
Returns: nothing

Called By:
- Restaurant::Modes()
-

Calls:
nothing
Function Logic description:
-This function is responsible for handling served orders
-In each time step it is called and do that :
-searchs for serving orders that its finished time equals currenttimestep for each type
-changes the statue of these orders from SRV to DONE and increases number of finished orders with each process
-searchs for the corresponding cook and make it unavailable
- returns NOrders,VOrders and GOrders by refernce to be used in display

- Restaurant::CountCooks (int& NCOOK_NUM, int& VCOOK_NUM, int& GCOOK_NUM)

Member of: Class Restaurant
Inputs:
NCOOK_NUM : Number availabe normal cooks
VCOOK_NUM : Number availabe VIP cooks
GCOOK_NUM : Number availabe Vegan cooks
Returns: nothing

Called By:
- Restaurant::Modes()
-

Calls:
nothing
Function Logic description:
-This function is responsible for counting available cooks in each timestep to be used in display

Section1:Hazem mahmoud abdo, Sec:1, B.N:26
- Restaurant::AssignOrder (int CurrentTimeStep,LinkedList<Cook*>&LastAssigned)

Member of: Class Restaurant
Inputs:
CuurentTimeStep
Pointer to the last assigned cook
Returns: nothing

Called By:
- Restaurant::Modes()
-

Calls:
nothing
Function Logic description:
-This function is responsible for handling waiting orders
-In each time step it is called and do that :
-It searches for first waiting orders.
-Decides which order to be served first and which cook will do that order depending on the priority of order(there is difference in priority among different orders types and there is difference in priority among VIP orders and each other).
-changes the statue of these orders from WAIT to SRV .
-changes the statue of these cooks from Avail to Unavail.
-calculates finished time of orders.
-If normal order waited for more than auto promotion time then promotes it to VIP.

Section2: Michael Aziz Faheem, Sec:2, B.N:7
- Restaurant::Cook_Injury(int CurrentTimeStep)

Member of: Class Restaurant
Inputs:
CurrentTimeStep : The current timestep.
Returns: nothing

Called By:
- Restaurant :: Modes(PROG_MODE mode)

Calls:
- Order:: getFinishedTime ()
- Order:: getServingTime()
- Order:: SetFinishedTime()
- Order::SetServingTime()
- Cook::GetStatue()
- Cook:: GetFinishedOrders()
- Cook:: GetSpeed()

Function Logic description:
This function is responsible for the cooks' injuries.
In each time step it is called and do that:
- it generates a random number and compares it with the cook probability.
-if the generated number was less than the injury probability then it searches for the first busy cook if any.
-it makes the first busy cook injured.
-changes the statue of these cooks from Unavail to Injured_working.
-changes the order's serving time and finish time based on the new speed after injury which is half of its original speed.