UML Diagrams: MottSt
Jen Yu, Brooke Jin, Bryan Chan

APCS2 Pd5

PEKING WONG

PekingWong

Driver file
import: cs1.Keyboard
java.io.*
java.util.*
main method only

SDish extends Order

private String dishName
private int dishTime
private ArrayList<String> names
SDish(): picks a random name for the dishName, dishTime is default to 1.

MDish extends Order

private String dishName
private int dishTime
private ArrayList<String> names
MDish(): picks a random name for the dishName, dishTime is default to 2.

LDish extends Order

private String dishName
private int dishTime
private ArrayList<String> names
LDish(): picks a random name for the dishName, dishTime is default to 2.

Waiter

private ArrayList<Customer> CustomersList
private ArrayList<Integer> tables
- - - - -
Waiter(): Constructs a waiter with an empty customer list & a full table list.
Order getNextOrder(Customer c): gets the next order of the customer's order list.
boolean assignTable(Customer c): assigns a table to the customer, if possible.
Customer findCust(int table): finds & returns a customer given the table, else return null.
void serve(Customer c, Order o): serves the order to the customer & removes it.
void chooseAction(): prints out prompt.
boolean checkCommands(String input): carries out command if valid, based on input.
void addCustomer(Customer c): adds the customer to Waiter's customers list.
Void removeCustomer(Customer c): removes the specified customer from list.

Customer

private String name
private int tableNum
private int VIPNum
ArrayList<Order> orders
- - - - -
Customer(): constructs a Customer with a default name, and random dishes in her/his orders ArrayList.
Customer(String name): sets name as param.
String toString(): returns string of the Customer's name & her/his pending orders.
boolean equals(Customer c): Returns true if customers sitting @ same table.
int compareTo(Customer other): Returns -1 if calling cust. has lower VIP# than other, 0 if the same & 1 if calling > other.
void addOrder(Order new): adds order.
int removeOrder(Order old): removes order.
void setTable(int num): sets table number.
Order findOrder(String name): finds an order given its name.
ArrayList<Order> getOrders(): Returns the ArrayList orders.
int getTable(): returns table #.
int getVIPNum(): returns the VIPnum of the customer.
String getName(): returns name.

Order

private String dishName
private int tableNum
- - - - -
Order(String s, int num): makes an order with dishName s and at table # num
String toString(): Returns the Table # & order name.
boolean equals(order name): T if names & tables are same.
void setTable(int t): sets table.
String getDishName(): gets the name of the dish.
int getTable(): gets table # of the order.

Kitchen

private Deque<String> pendingFoodList
private Queue<String> finishedFoodList
- - - - -
Kitchen(): initialize the deque and queue for usage.
void makeFood(): move food from pending list to finished list
void removeFirstP(): removes the first item from the pending list
void addFirstP(Order o): adds an item to the beginning of pending
void addLastP(Order o): adds an item to the back of pending.
void enqueueF(Order o): enqueues the order to finished food queue.
void dequeueF(Order o): dequeue the order from the fin. food queue.

Augmentation:

private int state, mood
void update(): updates the mood of the customer based on time elapsed.
int getState()
int getMood()