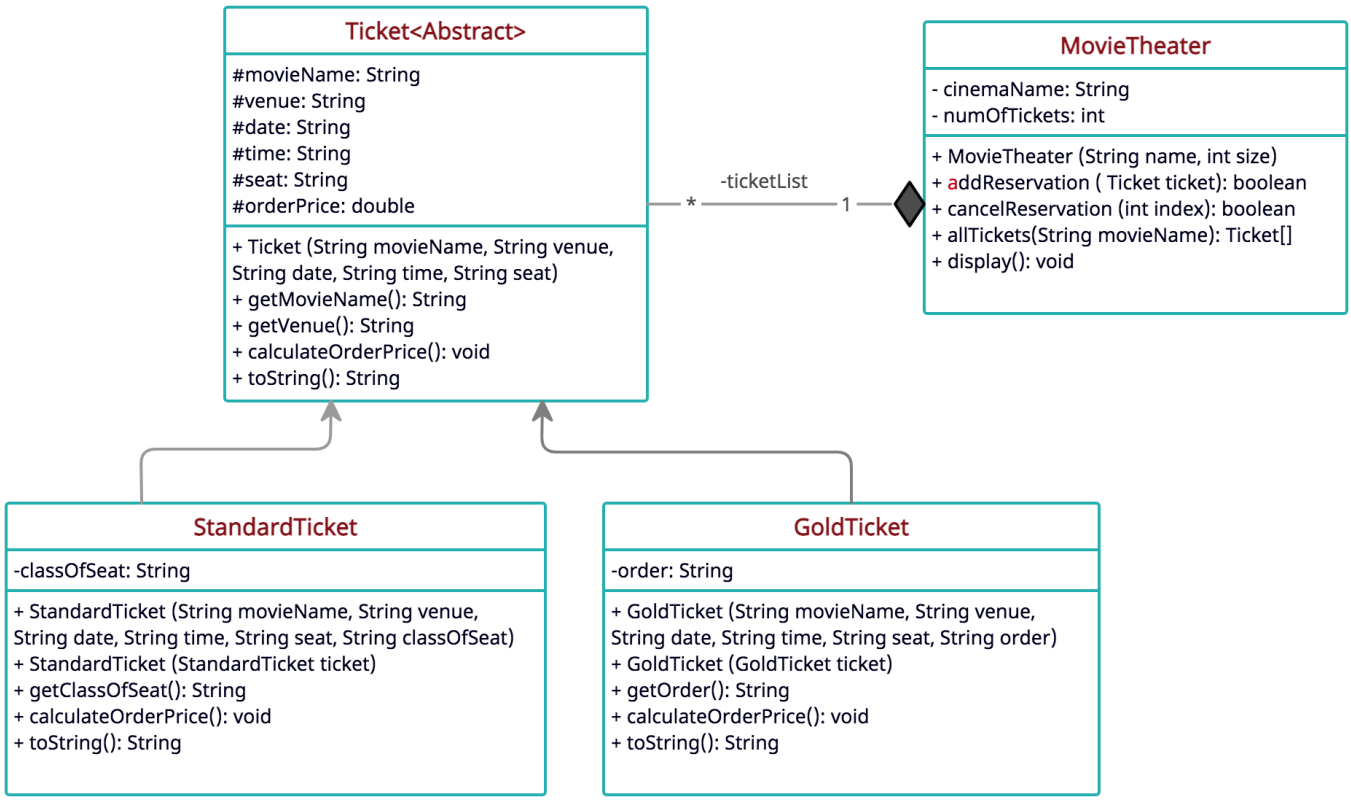
KING SAUD UNIVERSITY COLLEGE OF COMPUTER AND INFORMATION SCIENCES Computer Science Department		
CSC 113: Introduction to Programming II	**Sheet#1**	**2nd Semester 2020-2021**



Question#1: You should implement a program to manage the ticket reservations for a specific movie theater. Each movie theater offers two types of movie tickets, which are *standard ticket* and *gold ticket*. For the standard ticket, a movie theater usually offers two seating view options, which are standard view or premium view. While for the gold ticket, all seats have premium view and customers are offered a set of dishes that served directly to their seats. Given the above UML diagram, write the complete java implementation for all classes according to the following description:

Class Ticket (Abstract):

1. Attributes
    - **movieName:** A string to represent the movie name.
    - **venue:** A string to represent the location of the movie theater.
    - **date:** A string to represent the date according to the following format (dd/mm/yyyy).
    - **time:** A string to represent the time according to the following format (hh:mm).
    - **seat:** A string to represent the seat number according to the following format (L-N). The L represents the letter, which can range from A-L, while the N represents the number, which range from 1-9.
    - **orderPrice:** A double value to represent the price of the movie ticket.

2. Methods

- **Ticket(String movieName, String venue, String date, String time, String seat):** A constructor to initialize the movierName, venue, date, time, and seat with the received parameters.
- **getMovieName():** Returns the name of the booked movie.
- **getVenue():** Returns the venue of the booking movie theater.
- **calculateOrderPrice():** The total order price varies among different types of movie tickets. Hence, calculateOrderPrice method should be implemented as an abstract method.
- **toString():** Returns a string representation of Ticket object.

Class StandardTicket:

1. Attributes
   - **classOfSeat:** A string to represent the selected seating view, which can be either "Standard" view or "Premium" view.
2. Methods
   - **StandardTicket(String movieName, String venue, String date, String time, String seat, String classOfSeat)**: A constructor to initialize the movieName, venue, date, time, seat, classOfSeat with the received parameters. In addition, the constructor should call the calculateOrderPrice method to update the order price for a ticket.
   - **StandardTicket(StandardTicket ticket):** A copy constructor to initialize the attributes. In addition, the constructor should call the calculateOrderPrice method to update the order price for a ticket.
   - **getClassOfSeat():** Returns the class of seat based on the viewing options.
   - **calculateOrderPrice():** Calculates and sets the price of the ticket based on the class of seat. For the standard ticket the price is the same for any venue. However, the premium ticket has two different prices depending on the venue. The total order price should computed as follows:
     - If seating view is "Standard":
       - For any venue: price = 50 + 15% VAT.
     - If seating view is "Premium":
       - If the venue in the "Riyadh Park": price = 60 + 15% VAT.
       - If the venue in the "Red Sea Mall": price = 75 + 15% VAT.
   - **toString():** Returns a string representation of the standardTicket object.

Class GoldTicket:

1. Attributes
   - **order:** A string to represent the dish ordered by the customer. Movie theaters offer three different dishes, which are "Chicken Burger", "Mac N Cheese", or "Classic Nachos".
2. Methods
   - **GoldTicket(String movieName, String venue, String date, String time, String seat, String order):** A constructor to initialize the movieName, venue, date, time, seat, order with the received parameters. In addition, the constructor should call the calculateOrderPrice method to update the order price for a ticket.
   - **GoldTicket(GoldTicket ticket):** A copy constructor to initialize the attributes. In addition, the constructor should call the calculateOrderPrice method to update the order price for a ticket.
   - **getOrder():** Returns the dish name.
   - **calculateOrderPrice():** Calculates and sets the price of the order for the gold ticket based on the type of dish ordered by the customer. The total price of the order for the gold ticket should include: the ticket price, the dish price, and (15%) Value Added Tax (VAT). The price of the ticket is 150 R.S., however, the offered dishes have different charges which can be computed as follows:
     - If the dish name is "Chicken Burger": price = (150 + 60) + 15% VAT.
     - If the dish name is "Mac N Cheese": price = (150 + 50) + 15% VAT.
     - If the dish name is "Classic Nachos": price = (150 + 40) + 15% VAT.
   - **toString():** Returns a string representation of the goldTicket object.

Class MovieTheater:

1. Attributes
   - **cinemaName:** A string to represent the cinema name.
   - **numOfTickets:** Count the number of tickets that currently issued by this cinema.
2. Methods
   - **MovieTheater(String name, int size):** A constructor that initialize the cinemaName, numOfTicket, and creates a list of Ticket using the size specified as a parameter.
   - **addReservation(Ticket ticket):** Add the received ticket in the first empty location in the array list of tickets. Returns true if the addition was done successfully, and false otherwise.
   - **cancelReservation(int index):** Delete the ticket located at the received index, an then shifts the locations. Returns true if the deletion was done successfully and false if the received index is not correct [not in the range].

- **allTickets(String movieName):** Returns an array of all the reserved goldTicket objects that booked for a specific received movie name.
- **display():** Display all the information of the movie theater including the cinemaName, the number of reserved tickets, and all its issued tickets information.

Finally, implement **ReservationsTest** class with main method to do the following tasks:

1. Create a MovieTheater object, called **voxCinema** with the cinema name "**VOX**" which can hold a maximum of **20** tickets.
2. Add the following tickets to **voxCinema**:

Movie Name	Venue	Date	Time	Seat	Seat class/Dish name [depends on the type of created object]
Born A King	Riyadh Park	26/02/2021	10:00	J-5	Standard
Goodbye	Red Sea Mall	26/02/2021	15:00	H-5	Premium
Social Dilemma	Riyadh Park	27/02/2021	18:30	K-2	Chicken Burger
Born A King	Riyadh Park	28/02/2021	20:15	J-8	Classic Nachos

   Note: If the addition was successful, print a confirmation message that the ticket was added successfully, otherwise, print a message indicating that the ticket was not added.

3. Display all the tickets that issued by **voxCinema**.
4. Cancel reservation number 3 (which has index 2).
   Note: If the deletion was successful, print a confirmation message that the ticket was found and deleted successfully, otherwise, print a message indicating that the ticket was not found.
5. Get all gold tickets that belong to "Born A King" movie, and print all their information.
6. Display all the tickets that issued by **voxCinema** (after the update).

IMPORTANT NOTE: Make sure you use the same variable, method and class names as shown in the UML. Not doing so will affect your mark.