1)

Implement a shopping cart using the ArrayList class.

The file Item.java contains the definition of a class named Item that models an item one would purchase.

An Item has a name, price, and quantity (the quantity purchased).

The file Shop.java models shopping.

A. Fill in Shop Class

a. Create a cart - Declare and instantiate a variable cart to be an empty ArrayList.

b. Add item to cart - Fill in the statements in the loop to add an item to the cart.

Comments in the code indicate where these statements go.

B. Print out contents of the cart – After adding the cart:

a. Create a loop – Create a loop that gets each item from the cart.

i. Print each item

ii. Add up the total price – Add up the total price of each item [getPrice()] times the quantity of that

item [getQuantity()] that is in the cart.

b. Print out cart totals – Once the loop is complete, use System.out.println() to output the total price of all

the items in the cart.

2)

Develop an application that plays a simple dice game between the computer and the user. When the program runs, a loop should repeat 10 times. Each iteration of the loop should do the following:

Generate a random integer in the range of 1 through 6. This is the value of the computer’s die.

Generate another random integer in the range of 1 through 6. This is the value of the user’s die.

The die with the highest value wins. In case of a tie, there is no winner for that particular roll of the dice.

As the loop iterates, the program should keep count of the number of times the computer wins, and the number of times that the user wins. It should also display the results of each roll.

After the loop performs all of its iterations, the program should display who was the grand winner: the computer or the user.

3)

Create a simple application for the bus travelling service using

1. Book ticket for users/passengers on given routes via the various payment method.
2. Cancel the ticket booked by the users/passengers using the ticket details and the user credentials
3. Print the ticket via ticket details and the user credentials.
4. Update the passenger details on the booked ticket via ticket ID or E-mail ID and user credentials which are registered.

**Approach:** Initially, we need to set up a database in order to store the information of the busses and the booking. Here, the MySQL database is used. Initially, we need to set up the database. So, the following steps are followed to set up the database:

1. Initially, a database is created in MySQL database using the workbench.
2. After creating the database, multiple tables are created signifying multiple travel services where each table contains details of the passenger. The tables are created in the following way:
3. Now, since each table must contain the details of the passengers, each table has the following attributes:

Now, we need to create a connection between the database created above and the java program. In order to do this, the following steps are followed:

* 1. Initially, we need to collect the database and driver info in the class which we would be developing the application
  2. After initializing the database, we need to connect to it. So, we need to create a connection. The connection is created as
  3. After connecting, we should be able to execute SQL statements in order to get the data or update it or post it on the database. In order to execute the statements

Till now, we have initialized the database and created a connection between the database and the Java program. Now, we need to define all the methods based on functionality. The methods in the program are as follows:

1. **Book ticket:** In order to book a ticket, we first need to choose the route. This option is given to the user to choose a route among the set of predefined routes. For all the routes, users have to enter his details like name, age, mobile number, email, etc. Now, the user gets a list of available travels in the particular route. Based on the users choice, the data is stored in that respective table of the database created above. In order to do this, [switch-case](https://www.geeksforgeeks.org/switch-statement-in-java/) is used where every case is the choice of the travel provider.
2. **Cancel Ticket:** Similar to the above method, we need to implement the cancel functionality where users will be able to cancel the booked ticket. In order to do this, we need to first get the details of the users whose tickets need to be cancelled and this is validated with the mobile number and email ID of the user which is given during the input.
3. **Print Ticket:** Now, the print function is implemented. In order to print the ticket, the user details are taken, like the bus in which the ticket has been booked and the email ID is taken as the input to find the ticket details and print the details.
4. **Update Ticket:** In order to update the ticket, the user must have a booked ticket in the first place. Therefore, the email id of the user is taken as the input to verify if the ticket exists or not. After the email id is obtained, the search operation is performed on the database to search the ticket and then the new updated details are taken as the input which is then updated in the database.

4)

Create a class Movie with the following private member variables.

* String movieName
* String movieCategory
* int ticketCost

Include appropriate getters and setters method in Movie class.  
Write the following method in the Movie class:

public int calculateTicketCost(String circle)--This method should take circle as argument. Return 0 when any one of the following condition met :

* If circle is gold and movie category is 2D set ticket cost Rs.300
* If circle is gold and movie category is 3D set ticket cost Rs.500
* If circle is silver and movie category is 2D set ticket cost Rs.250
* If circle is silver and movie category is 3D set ticket cost Rs.450

Return -1 if category is invalid.  Return -2 if circle is invalid and -3 if both are invalid.

Create a Main class to get the input and to display the output

Get the input for Movie using the method - public static Movie getMovieDetails(). Invoke this method from the main method and store the Movie object.

Get the input for circle using the method - public static String getCircle().  Invoke this method from the main method and store the circle value.

Next, invoke the calculateTicketCost method in Movie by passing this circle and calculate the ticket cost and set to the ticketCost attribute and print the output.

If the return from calculateTicketCost is zero, the output should be be  "The ticket cost is=<ticket cost>".

If it returns -1, the output should be "Sorry there is no <category> type of category in theater."

If the return is -2, the output should be "Sorry!!! Circle is Invalid."

If the return is -3, the output should be "Sorry!!! Both circle and category are Invalid."

**Sample Input :**

Enter the movie name:  
VIP  
Enter the movie category:  
2D  
Enter the circle:  
gold

**Sample Output :**

Movie name = VIP  
Movie category = 2D  
The ticket cost is = 300

5)

In the "ABC" hotel you have the data of the guests, for each guest you have:  
name, type of room (1,2,3,4,5) and the number of days it was occupied. A discount is made  
10% if the client stays more than 5 days, 15% if he stays more than 10 days and 20% if  
stays more than 15 days; according to the type of room you have the following rates  
Room type Daily rate  
1. $ 120.00  
2. $ 155.00  
3. $ 210.00  
4. $ 285.00  
5. $ 400.00  
You are asked to prepare the program that reads the details of the guests. The hotel has 20 rooms in total, 4 of each type. You must validate if the room you want is available from otherwise give him the option to choose another, you must also validate if the hotel is full, which you must express with a message.  
You should have an option to prepare a report as follows:  
Guest Name Days Subtotal Rate Total Discount  
Total guests  
Total rate  
You'll need to use overloaded object arrays and constructors.

---------------------------------------------------------------------------------------------------

6)

Develop a gaming application that picks a random number from 50-150. The user keeps guessing as long as their guess is wrong, and they've guessed less than 7 times. If their guess is higher than the number, say "Too high." If their guess is lower than the number, say "Too low." When they get it right, the game stops. Or, if they hit seven guesses, the game stops even if they never got it right.

7)

Develop an application for managing Employee details using List

Find employees aged over 40

Find employees from the USA

Sort employees by country

8)

Develop an application for Stock maintenance using Map

Find the highest stock price

Find the average stock price

Remove companies with stock price below 200

9)

Develop an application for maintaining salary of the employees using Set

Find the count of the employees getting salary greater than 30000

Find highest and lowest salary in set

10)

“ABC Restaurant” owner has engaged you to write a Java program to take the restaurant customer order.

The food menu of the restaurant is available to the customer, This program must allow customer to decide how many items he/she wish to order.

Then, it will let the customer to choose the each item according to his/her preference. Upon completion of the ordering, the program will display the total amount of the order to the customer. Then, the program must ask the customer whether to do another set of ordering.

In your code, you must create a method to process the item choice and return the price of the item.