Name of the Course : Java for beginners: Step–by–step hands-on guide to Java

Level : Difficult

Tool Stack : Encapsulation, basic java programming and access specifires

Problem Statement : Tom want develop application for book movies tickets. Develop Ticket Calculation application for above scenario.

Description : Ticket Calculation

Create a class Ticket with the following private variables

int ticketid;

int price;

static int availableTickets;

Include getters and setters methods in the Ticket class.

AvailableTickets should hold only positive value. Zero and negative values are not allowed.(This logic should be checked inside the corresponding setter method)s

Write the following method in the Ticket class:

public int calculateTicketCost(int nooftickets) —this method should check the ticket availability,

If the tickets are available, reduce the nooftickets from availableTickets and calculate the total

amount as nooftickets\*price and return the total amount. If the tickets are not available,

this method should return -1.

Write a main method in the Main class to test the application.

**Code:**

**import** java.util.Scanner;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

**int** i;

Ticket t=**new** Ticket();

Scanner br=**new** Scanner(System.***in***);

System.***out***.println("Enter no of bookings:");

**int** nob= br.nextInt();

System.***out***.println("Enter the available tickets:");

**int** availableTickets=br.nextInt();

Ticket.*setAvailableTickets*(availableTickets);

**for**(i=1;i<=nob;i++)

{

System.***out***.println("Enter the ticketid:");

**int** ticketid= br.nextInt();

t.setTicketid(ticketid);

System.***out***.println("Enter the price:");

**int** price= br.nextInt();

t.setPrice(price);

System.***out***.println("Enter the no of tickets:");

**int** nooftickets= br.nextInt();

**int** totalamount;

System.***out***.println("Available tickets: "+t.*getAvailableTickets*());

totalamount=t.calculateTicketCost(nooftickets);

**if**(totalamount!=-1)

{

System.***out***.println("Total amount:"+totalamount);

System.***out***.println("Available ticket after booking:"+t.*getAvailableTickets*());

}

}

}

}

**public** **class** Ticket {

**private** **int** ticketid;

**private** **int** price;

**private** **static** **int** *availableTickets*;

**static** **int** *count*=1;

**public** **int** getTicketid() {

**return** ticketid;

}

**public** **void** setTicketid(**int** ticketid) {

**this**.ticketid = ticketid;

}

**public** **int** getPrice() {

**return** price;

}

**public** **void** setPrice(**int** price) {

**this**.price = price;

}

**public** **static** **int** getAvailableTickets() {

**return** *availableTickets*;

}

**public** **static** **void** setAvailableTickets(**int** availableTickets) {

**if**(availableTickets>0)

{

Ticket.*availableTickets* = availableTickets;

}

**else**

{

*count*=0;

}

}

//method

**public** **int** calculateTicketCost(**int** nooftickets)

{

**int** avt=*getAvailableTickets*();

avt=avt-nooftickets;

**int** price=getPrice();

*setAvailableTickets*(avt);

**if**(*count*==1)

{

price=price\*nooftickets;

setPrice(price);

**return** price;

}

**else**

**return** -1;

}

}

Junit Testing

**import** **static** org.junit.jupiter.api.Assertions.\*;

**import** java.io.IOException;

**import** org.junit.jupiter.api.Test;

**import** main.java.yaksha.TestUtils;

**import** main.java.yaksha.Ticket;

**class** MainTest {

@Test

**void** testCalculateTicketCost() **throws** IOException {

Ticket t=**new** Ticket();

Ticket.*setAvailableTickets*(20);

t.setTicketid(10);

t.setPrice(100);

*assertEquals*(500,t.calculateTicketCost(5));

}

}

Test Data1

Enter no of bookings:

1

Enter the available tickets:

20

Enter the ticketid:

10

Enter the price:

100

Enter the no of tickets:

5

Available tickets: 20

Total amount:500

Available ticket after booking:15

Test Data2

Enter no of bookings:

1

Enter the available tickets:

50

Enter the ticketid:

100

Enter the price:

100

Enter the no of tickets:

5

Available tickets: 50

Total amount:500

Available ticket after booking:45

Learning outcome: Participant could able to learn how to use encapsulation and protected access specifires.