1.Miraj Cinemas partners with BookMyShow to increase attendance and revenue, As a software consultant for BookMyShow, assist Miraj Cinemas in creating software that will include functionality to determine the cost of a ticket to be purchased and generate a ticket ID.

**Component Specification: MovieTickets**

|  |  |  |
| --- | --- | --- |
| **Type(Class)** | **Attributes** | **Methods** |
| **MovieTickets** | String movieName  String   screenNumber  int noOfTickets  String mobileNumber  String time  String modeOfPayment | Necessary Getters, setters, and six argument constructors are provided as a part of the code skeleton. |

**Functional Requirement 1: Extract the details of MovieTickets and create an object of the MovieTickets class**

|  |  |  |
| --- | --- | --- |
| **Type(Class)** | **Attributes** | **Responsibilities** |
| **UserInterface** | public static MovieTickets **extractDetails**(String movieDetails) | This method accepts **movieDetails** separated by colon as an argument and should extract the properties of the **MovieTickets**from the argument by parsing. Set these values to the **MovieTickets**object and return this object. |

**Functional Requirement 2: Calculate the total amount to be paid by the customer.**

|  |  |  |
| --- | --- | --- |
| **Type(Class)** | **Attributes** | **Responsibilities** |
| **MovieTickets** | public double **calculatePrice**() | This method is used to calculate the total price to be paid by the customer and return the same.  Kindly refer to the below table for the price of each screen and the convenience charge for each transaction respectively.  ***Condition:***   * *If the****screenNumber****does not match the below table then, return 0.* * ***screenNumber****is case sensitive.* * *If the****noOfTickets****is less than or equal to zero then, return 0.* |

|  |  |  |
| --- | --- | --- |
| **Screen Number** | **Price** | **Convenience Charges** |
| S1  S2  S3  S4  S5 | Rs.280  Rs.250  Rs.520  Rs.400  Rs.180 | Rs.35  Rs.35  Rs.35  Rs.35  Nil |

**Formula to calculate the total amount to be paid:**

**Amount to be paid = no.of.tickets \* TicketPrice+ convenience charge**

Eg: Let the screen number be S3, movie price be Rs.520, noOfTicket be 4 nos,

Amount to be paid =4\*520+35=Rs.2115.0

**Functional Requirement 3: Generate the ticket Id**

|  |  |  |
| --- | --- | --- |
| **Type(Class)** | **Attributes** | **Responsibilities** |
| **MovieTickets** | public String**generateTicketId()** | TicketId will be generated based on the movieName, screenNumber and noOfTickets.  ***Condition:***   * *If****screenNumber****or****noOfTickets****are invalid then, return "****Invalid movie details****"* |

**For example**

Let Movie Name be Love Today, Screen Number be S1, and number of tickets be 15, then

**TicketId:  LvS1N15**

Lv - First 2 consonants of the movie name

S1 - represents the Screen number

N15 - represents the Number of tickets

**The main method is excluded from the evaluation. You are free to write your own code or add lines of code to check the correctness of the functionalities**

**Note:**

* In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user, and the rest of the text represents the output.
* Ensure to follow the object-oriented specifications provided in the question.
* Ensure to provide the names for classes, attributes, and methods as specified in the question.
* Adhere to the code template, if provided.

**Sample Input/Output -1**

Enter the details

**Love Today:S1:15:9958456545:10.00Pm:Gpay**

Ticket Details

Movie Name: Love Today

Screen Number: S1

Number of tickets: 15

Show Timing: 10.00 Pm

Ticket Cost: 4235.0

TicketId:  LvS1N15

Thank you! Your payment received via Gpay, Your movie tickets are confirmed.

**Sample Input/Output -2**

Enter the details

**Frozen:S5:7:9957856545:2.00Pm:PayTM**

Ticket Details

Movie Name: Frozen

Number of tickets: 7

Screen Number: S5

Show Timing: 2.00PM

Ticket Cost: 1260.0

TicketId: FrS5N7

Thank you! Your payment received via PayTM, Your movie tickets are confirmed.

**Sample Input/Output -3**

Enter the details

**Bigil:S6:12:8545456465:12.00PM:Card**

Invalid movie details

**Sample Input / Output -4**

Enter the details

**Bigil:S4:-4:8545456465:12.00PM:Card**

Invalid movie details

2. The **"Malgudi Express"** was supposed to be launched as part of the ongoing "Azadi Ka Amrit Mahotsav" celebration of India's 75th year of independence. It's a beautiful train ride along the Western Ghats through typical southern Indian scenery. You have been approached as their software consultant to develop software that will implement the functionality to calculate the total ticket charge that the passenger will be charged.

**Component Specification: TicketInfo**

|  |  |  |
| --- | --- | --- |
| **Type (class)** | **Attributes** | **Methods** |
| **TicketInfo** | String name  int noOfTickets  String coach  String mobileNumber  String insurance | Necessary Getters, setters and five argument constructor are provided as a part of the code skeleton |

**Functional Requirement 1:** **Extract the ticket details and create an object of the TicketInfo class.**

|  |  |  |
| --- | --- | --- |
| **Type(class)** | **Methods** | **Responsibilities** |
| **UserInterface** | public static **TicketInfo extractDetails**(String ticketDetails) | This method accepts passenger details as an argument and extracts the properties of the TicketInfofrom the argument. Set these values to the TicketInfo object and return this object. |

**Functional Requirement 2:** **Calculate the cost of the ticket, including the service charge and GST.**

|  |  |  |
| --- | --- | --- |
| **Type(class)** | **Methods** | **Responsibilities** |
| **TicketInfo** | public double **calculateTicketCost**() | This method is used to calculate the total cost to be paid by the passenger.  If the preferred coach is "**FirstAC**", there is a 5% GST and a 4.5% service charge.   If the preferred coach is "**SecondAC**", there is a 5% GST and a 4.5% service charge.  If the preferred coach is "**ThirdAC**", there is a 4% GST and a 4.5% service charge.  If the preferred coach is "**Sleeper"**, there is a 2% GST and a 4.5% service charge.  If the preferred coach is "**Unreserve**", there is no GST and service charge.  ***Condition:****The GST rate varies depending on the coach. All coaches pay the same service fee of 4.5%.*  *The preferred****coach****is case-insensitive.*  *If the preferred****coach****does not match the above cases, return -1.* |

|  |  |  |
| --- | --- | --- |
| **Preferred Coach** | **Cost** | **GST(%)** |
| FirstAC | 2200 | 5 |
| SecondAC | 1700 | 5 |
| ThirdAC | 1200 | 4 |
| Sleeper | 520 | 2 |
| Unreserve | 200 | None |

**Formula to calculate Ticket Cost:**

Ticket Cost=noOfTickets\*(cost+(cost\*5% GST)+(cost\*4.5% Service charge))

**Example:** Let cost be 2400,

Ticket Cost=2400+(2400\*(5/100))+(2400\*(4.5/100))

Ticket Cost=2400+120+108=2628.0.

**Functional Requirement 3:** **Calculate the total booking cost, including insurance if required.**

|  |  |  |
| --- | --- | --- |
| **Type(class)** | **Methods** | **Responsibilities** |
| **TicketInfo** | public double  **calculateTotalBookingCost**() | This method calculates the total booking cost to be paid by including the insurance charge (if required) with the **Ticket Cost**.  If the 'insurance' field is set to "**yes**", the total booking cost is confirmed by including the insurance charge.  If the 'insurance' field is set to "**no**," the insurance charge is not included.  ***Condition:***  *The insurance charge is Rs. 49 per person.*  *The****insurance****field is case-insensitive.*  *If the****insurance****field is set to something other than "****yes****" or "****no****", then return -1.* |

**Formula to calculate Ticket Booking Cost: (If Insurance required)**

Total Cost=noOfTickets\*(Ticket Cost+49).

**Example:**

Total Cost=2628.0+49=2677.0

**The main method in the UserInterface class is excluded from evaluation. You are free to write your own code in the main method to invoke the business methods to check its correctness.**

**Note:**

* In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the rest of the text represents the output.
* Ensure to follow the object-oriented specifications provided in the question.
* Ensure to provide the names for classes, attributes, and methods as specified in the question.
* Adhere to the code template, if provided.

**Sample Input / Output 1:**

Enter the passenger details

**Alex:2:FirstAC:8879678456:yes**

Passenger Details

Passenger Name: Alex

Number of Tickets: 2

Coach: FirstAC

Mobile Number: 8879678456

Insurance Required: yes

Total Booking Charge: Rs. 4916

**Sample Input / Output 2:**

Enter the passenger details

**Mary:2:unreserve:8879789656:no**

Passenger Details

Passenger Name: Mary

Number of Tickets: 2

Coach: unreserve

Mobile Number: 8879789656

Insurance Required: no

Total Booking Charge: Rs. 400

**Sample Input / Output 3:**

Enter the passenger details

**Leena:4:TwoTierAC:8890567890:yes**

Invalid Passenger details

**Sample Input / Output 4:**

Enter the passenger details

**David:5:unreserve:9884455661:y**

Invalid Passenger details

## 3. Orbiton Technologies - Map

Orbiton technologies conducts interviews and selects candidates for further interview process. The recruitment process consists of three rounds. A first round is conducted for all the candidates who have applied for the interview. Candidates are chosen for the second round based on their performance in the first round. Some candidates are shortlisted directly for the final interview based on certain criteria. You, being the software developer, develop a Java program to shortlist the candidates based on the requirement.

**Component Specification: CandidateMain Class**

|  |  |  |
| --- | --- | --- |
| **Type (Class)** | **Attributes** | **Methods** |
| **CandidateMain** | Map<String, Double> candidateMap | Getter and setter methods for the attribute are included in the code skeleton. |

***Note:****key : candidateId and value: markScored for candidateMap attribute.*

**Requirement 1: Find the candidate mark based on the given candidateId**

|  |  |  |
| --- | --- | --- |
| **Type (Class)** | **Methods** | **Responsibilities** |
| **CandidateMain** | public double **findCandidateMark**(String candidateId) | This method accepts candidateId as argument. If the candidateId is found in the Map, return the candidate's markScored. Else return -1.  ***Condition:****candidateId is a case-sensitive* |

**Requirement 2: Find the number of candidates shortlisted for further interview process.**

|  |  |  |
| --- | --- | --- |
| **Type (Class)** | **Methods** | **Responsibilities** |
| **CandidateMain** | public int **findCountOfCandidatesShortlisted**() | This method counts the number of candidates shortlisted for further interview process based on the below condition.  ***Condition:****Candidates with markScored greater than or equal to 70 are shortlisted for further interview process.* |

**Requirement 3: Filter the candidates shortlisted for final interview.**

|  |  |  |
| --- | --- | --- |
| **Type (Class)** | **Methods** | **Responsibilities** |
| **CandidateMain** | public List<String> **findCandidatesShortlistedForFinalInterview**() | This method filters the candidates (candidateMap) shortlisted for final interview based on the below condition, add the candidate ids to the list and returns the list."  ***Condition:****candidates with markScored greater than or equal to 9****0****are directly shortlisted for the final interview.* |

**You are provided with the main method as code template and it is excluded from evaluation.**

**Note:**

* In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user, and the rest of the text represents the output.
* Ensure to follow the object-oriented specifications provided in the question description.
* Ensure to provide the names for the classes, attributes, and methods as specified in the question description.
* Adhere to the code template, if provided.

**Sample Input/Output 1:**

Enter the number of records to be added:

**6**

Enter the candidate records (candidateId : markScored):

**ORB101:75.7**

**ORB102:79.5**

**ORB103:95.3**

**ORB104:67**

**ORB105:92**

**ORB106:54**

Enter the candidate id to be searched

**ORB103**

95.3

Number of candidates shortlisted for further interview process:

4

Candidate id of the candidates shortlisted to directly attend the final interview:

ORB103

ORB105

**Sample Input/Output 2:**

Enter the number of records to be added:

**6**

Enter the candidate records (candidateId : markScored):

**ORB101:87.9**

**ORB102:65.8**

**ORB103:61.2**

**ORB104:98**

**ORB105:84**

**ORB106:54**

Enter the candidate id to be searched

**ORB108**

ORB108 is an invalid candidate id

Number of candidates shortlisted for further interview process:

3

Candidate id of the candidates shortlisted to directly attend the final interview:

ORB104

## 4. Puddles -A Piggy Bank

Puddles a digital piggy bank, is a tool that can help you save money.  You being the software developer, develop a Java program based on the requirement.

**Component Specification: PiggyBank**

|  |  |  |
| --- | --- | --- |
| **Type(Class)** | **Attributes** | **Responsibility** |
| **PiggyBank** | private Map<Integer, Integer> moneyMap | Getter and setter methods for the attribute are included in the code skeleton. |

***Note****Key: Denomination, Value: countOfDenomnations in moneyMap attribute.*

**Requirement 1: Calculate the total amount of money in the piggy bank.**

|  |  |  |
| --- | --- | --- |
| **Type(Class)** | **Method** | **Responsibility** |
| **PiggyBank** | public double **calculateMoneyAvailable**() | This method calculates the sum of money in the piggy bank and returns the result.  ***Condition:***   * ***If the sum is zero, return -1.*** |

**Formula to calculate money available:**

Money available = Sum of the product of all available denominations and its count

**Example: 10:1 1:1 5:1 20:1 2:1**

**MoneyAvailabe=10\*1+1\*1+5\*1+20\*1+2\*1**

**=38**

**Requirement 2: Retrieve the available denominations and return the sorted list of denominations**

|  |  |  |
| --- | --- | --- |
| **Type(Class)** | **Method** | **Responsibility** |
| **PiggyBank** | public List<Integer> **retrieveCurrencyDenominations**() | This method has to return the sorted list that contains all the available denominations. |

**Requirement 3: Calculate money based on the denomination and return the total value.**

|  |  |  |
| --- | --- | --- |
| **Type(Class)** | **Method** | **Responsibility** |
| **PiggyBank** | public double **calculateMoneyBasedOnDenomination**(int denomination) | This method takes denomination as a parameter and returns total money available based on denomination.  ***Condition:***   * *If the denomination is present in the****moneyMap****, return the total value of the denomination, else****return -1****.* |

**Formula to calculate money based on the denominations**

CalculateMoneyBasedOnDenomination = Product of given a denomination and its count

**Example: 10:1 1:1 5:1 20:1 2:1**

**For 20,**

**CalculateMoneyBasedOnDenomination=20\*1**

**=20**

**You are provided with the main method as a code template and it is excluded from evaluation.**

**Note:**

* In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user, and the rest of the text represents the output.
* Ensure to follow the object-oriented specifications provided in the question description.
* Ensure to provide the names for the classes, attributes, and methods as specified in the question description.
* Adhere to the code template, if provided.

**Sample Input/Output - 1**

Enter the total number of money to added

**5**

Enter your piggy bank entries (denomination: count)

**10:1**

**1:1**

**5:1**

**20:1**

**2:1**

Total money in the Piggy bank: 38.0

Available Denominations:

1

2

5

10

20

Enter the denomination

**20**

Total amount of the denomination 20 is 20.0

**Sample  Input/Output -2**

Enter the total number of money to added

**3**

Enter your piggy bank entries (denomination: count)

**100:5**

**500:2**

**2000:1**

Total money in the Piggy bank: 3500.0

Available denominations:

100

500

2000

Enter the denomination

**100**

Total amount of the denomination 100 is 500.0

**Sample Input/Output -3**

Enter the total number of money to added

**2**

Enter your piggy bank entries (denomination: count)

**5:0**

**10:0**

No money available

**Sample Input/Output -4**

Enter the total number of money to added

**3**

Enter your piggy bank entries (denomination: count)

**5:100**

**2:6**

**10:6**

Total money in the Piggy bank: 572.0

Available denominations:

2

5

10

Enter the denomination

**200**

Invalid denomination