

FA1 Hands-On

02:11:33

Duration: 120 minutes**Max Marks: 15****c3RhdmFuLnBhdGls****Instructions to candidates:**

1. Follow the class diagram strictly. Read the problem statement, examples and the other details provided carefully and implement the solution
2. Code submitted with compilation errors may not get evaluated
3. DO NOT add any new method apart from the one given in the file where you write the solution
4. DO NOT alter the method name or the argument list of the method that is provided to you
5. DO NOT write codes that result in infinite loops/infinite recursive calls, because it just won't work!
6. The test cases provided are a subset of test cases that will be used to test your code

Guidelines:

1. You (Trainee) should not tamper/modify the default folder structure provided. Any changes done to default folder structure will result in solution scripts not being evaluated.
2. You (Trainee) should save the code frequently.
3. The verify option checks the code for structural and logical correctness as per the requirements given in the problem statement. Please ensure that your code is structurally and logically correct according to the given problem statement before submitting.
4. Submission is a two step process. You (Trainee) should click **Submit** and **Finish Test** for considering the solution.

File Edit Selection View Go Run Terminal Exam Help

EXPLORER Reference Question X Customer.java

PROJECT

QP

images

Reference

index.html

To_Trainees

src

dsausingj...

progusingj...

Customer...

Digitalte...

MusicOrd...

Tester.java

GUIDELINES:

1. You (Trainee) should not tamper/modify the default folder structure provided. Any changes done to structure will result in solution scripts not being evaluated.

2. You (Trainee) should save the code frequently.

3. The verify option checks the code for structural and logical correctness as per the requirements given in the problem statement. Please ensure that your code is structurally and logically correct according to the given problem statement before submitting.

4. Submission is a two-step process. You (Trainee) should **Submit** and **Finish Test** for considering the solution scripts for evaluation.

1. You (Trainee) can **Submit** multiple times. The last submitted code would be considered for evaluation
2. There is no **Auto Submission** feature and hence you (Trainee) must finally submit the scripts mandatorily by clicking on **Finish Test**.
3. Clicking on **Finish Test** will be considered as the end of assessment. You will not be allowed to change code once you click on Finish Test.

5. In case of any issue in submission, contact the SPOC(s) assigned for further steps.

NOTE: NOT adhering to the above instructions and guidelines may lead to reduction in your score even if the code is executing without any errors

Few errors which might result in mark reduction:

1. Removal/modification of package statements

02:11:25

c3RhdmFuLnpBhdGls

Activate Windows

Go to Settings to activate Windows

Edit Selection View Go Run Terminal Exam Help

PLORER Reference Question x Customer.java

02:11:21

Few errors which might result in mark reduction:

1. Removal/modification of package statements
2. Infinite loops/infinite recursive calls in the code
3. Class diagram violation
4. Improper constructor prototype or definition

Question1: Object Oriented Programming:
[10 Marks]

Problem Description:

Digital Arts is a company which provides digital content like music to various customers and wants to automate the ordering process of digital items. Implement the class diagram given below to achieve the same.

Class Diagram:

```
classDiagram DigitalItemOrder "1..> Customer : -customer
```

Activate Windows
Go to Settings to activate Windows

NjA2YWNlODQwM2VmNDMzZTY5ZjcwYzUwIHN0YXZhbi5wYXRpbA-- 02:11:21

Fullscreen

Edit Selection View Go Run Terminal Exam Help

KPLORER

PROJECT

- QP
- images
- Reference
- index.html
- To_Trainees
- src
- dsusingj...
- prgusingj...
- Customer...
- Digitalite...
- MusicOrd...
- Tester.java

PENDENCIES

project_9d5f...

Reference Question **Customer.java**

Digital Arts is a company which provides digital content like music to various customers and wants to ordering process of digital items. Implement the class diagram given below to achieve the same.

02:11:17

Class Diagram:

DigitalItemOrder

- customer : Customer
- itemName : String
- orderPrice : double
- orderId : String
- counter → static : int
- +DigitalItemOrder(customer : Customer, itemName: String)
- + setOrderId(orderId : String) : void
- + setOrderPrice(orderPrice : double) : void
- + getItemName() : String
- + getCustomer() : Customer
- + getOrderId() : String
- + getOrderPrice() : double
- + generateOrderId() : void
- + calculateOrderPrice() : void → abstract

Customer

- custName : String
- emailId : String
- age : int
- +Customer(custName : String, emailId : String, age : int)
- + validateCustomerDetails() : Boolean
- + getCustName() : String
- + getEmailId() : String
- + getAge() : int
- + toString() : String

c3RhdmFuLnBhdGIs

Activate Windows

NjA2YWNlODQwM2VmNDMzZTY5ZjcwYzUwJHN0YXZbbi5wYXBnbA= - 02-11-18

```
graph LR; DigitalItemOrder --> Customer
```

File Edit Selection View Go Run Terminal Exam Help

Fullscreen

02:11:13

c3RhdmFuLnBhdGls

Customer.java

```
+ getOrderId() : String  
+ getOrderPrice() : double  
+ generateOrderId() : void  
+ calculateOrderPrice() : void → abstract  
+ toString() : String
```

MusicOrder

```
- musicFormat : String  
- bitRate : int  
- durationInSec : int  
- musicFormatAvailable → static : String[]  
- bitRateAvailable → static : int[]  
- bitRateCost → static : double[]  
+ MusicOrder(customer : Customer,  
itemName : String, musicFormat : String,  
bitrate : int, durationInSec: int) :  
DigitalItemOrder(customer, itemName)  
+ calculateOrderPrice() : void  
+ validateBitRate() : Integer  
+ toString() : String
```

Notes:

NjA2YWNlODQwM2VmNDMzZTY5ZjgwYzUwJHNnVYzbh5wVX8phA-- 02-11-14

You're being proctored!

File Edit Selection View Go Run Terminal Exam Help

Fullscreen

EXPLORER

PROJECT QP

- images
- Reference
- index.html
- To_Trainees
- src
- dsusingj...
- prgusingj...
 - Customer...
 - Digitalite...
 - MusicOrd...
 - Tester.java

Notes:

02:11:10

Customer.java

toString(): String

• Do not include any extra instance/static variables and instance/static methods in the given classes.

• Case **in-sensitive** comparison is required to be done unless specified.

• Do not change any value or case of the given variables.

• Read notes and examples for better understanding of the logic.

Implementation Details:

Class Name	Implementation Details
Customer	Partially implemented
DigitalItemOrder	Partially implemented
MusicOrder	Partially implemented

Customer class:

validateCustomerDetails():

- This method validates the customer details and returns Boolean based on the below logic:
 - If the **custName** contains at least 4 characters and **age** is more than 18 and **emailId** ends with either **.com** or **.edu**, then this method returns **true**.

Activate Windows

Nja2YWNlODQwM2VmNDMzTY5ZjcwYzUwIHN0YZZhbiswYYRohA= - 02:11:10

File Edit Selection View Go Run Terminal Exam Help Fullscreen

EXPLORER

PROJECT

QP

> images

> Reference

index.html

To_Trainees

src

> dsusingj...

progusingj...

Customer...

Digitalite...

MusicOrd...

Tester.java

REFERENCE

Customer.java

DigitalItemOrder

MusicOrder

Partially implemented

Partially implemented

Customer class:

validateCustomerDetails():

- This method validates the customer details and returns Boolean based on the below logic:
 - If the **custName** contains at least 4 characters and **age** is more than 18 and **emailId** ends with either **.com** or **.edu**, then this method returns **true**
 - Otherwise, return **false**

Note: Perform **case-sensitive** comparison

Example: If the **custName** is **Bobby**, **age** is 19 and **emailId** is **bob@xyz.com**, then this method returns **true**

DigitalItemOrder class:

generateOrderId():

- This method auto-generates and sets the **orderId(String)**
- The **orderId** would be prefixed with the first character of the **itemName(String)** followed by the auto-generated value starting from 1001
- The auto-generated value would be incremented by one for the next **orderId**

02:11:06 c3RhdmFuLnBhdGls

NJA2YWNlODQwM2VmNDMzZTY5ZjcwYzUwIHN0YyZhbISwY3RpbA= 02:11:06

File Edit Selection View Go Run Terminal Exam Help Fullscreen

EXPLORER PROJECT QP images Reference Question x Customer.java

DigitalItemOrder class:

generateOrderId():

- This method auto-generates and sets the **orderId**(String)
- The **orderId** would be prefixed with the first character of the **itemName**(String) followed by the auto-generated value starting from 1001
- The auto-generated value would be incremented by one for the next **orderId**
- Use static variable **counter** appropriately to implement the auto-generation logic

Note: Perform **case-sensitive** operations

Example: If the **itemName** is **Unforgiven**, then the first **orderId** would be **U1001**, second **orderId** generated would be **U1002** if the **itemName** is **numb**.

MusicOrder class:

musicFormatAvailable:

- This is a static array (String []), where each of the elements would be the **musicFormat**(String)
- The initial values of **musicFormatAvailable** are given below:

```
musicFormatAvailable | {"mp3", "wav", "flac"}
```

File Edit Selection View Go Run Terminal Exam Help Fullscreen

EXPLORER PROJECT QP images Reference index.html To_Trainees src dsusingj... progusingj... Customer... Digitalite... MusicOrd... Tester.java

02:10:59

MusicOrder class:

musicFormatAvailable:

- This is a static array (String []) , where each of the elements would be the `musicFormat(String)`
- The initial values of **musicFormatAvailable** are given below:

musicFormatAvailable	{ "mp3", "wav", "flac" }
-----------------------------	--------------------------

Note:

- This array is supplied. Hence no need to code
- Do not change the **CASE** of the elements in the array

bitRateAvailable:

- This is a static array(int[]) that contains the `bitRate (int)available`
- The initial values of **bitRateAvailable** are given below:

bitRateAvailable	{192, 128, 256}
-------------------------	-----------------

Activate Windows
Get the Software Protection License Key

File Edit Selection View Go Run Terminal Exam Help

Fullscreen

EXPLORER

PROJECT

- QP
- images
- Reference
- index.html
- To_Trainees
- src
- dsusingj...
- progusingj...
 - Customer...
 - Digitalite...
 - MusicOrd...
 - Tester.java

Reference Question x Customer.java

Note:

- This array is supplied. Hence no need to code

bitRateCost:

- This is a static array(double[]) that contains the cost (double) for each bitRate available
- This array has one to one correspondence with **bitRateAvailable** array
- The initial values of **bitRateAvailable** are given below:

bitRateCost	{15.0,10.0,20.0}
-------------	------------------

Note:

- This array is supplied. Hence no need to code

validateBitRate():

- This method validates the **bitRate**(int) and returns the *index*(Integer) based on the following logic:
 - Check if **bitRate** is present as one of the elements in **bitRateAvailable** array
 - If it is present, return the corresponding *index* position of the **bitRate** in **bitRateAvailable** array

Activate Windows
NJA2YWNlODQwM2VmNDMzZTY5ZjcwYzUwIHN0Y3ZhNcvYXBaA-- 02-10-57

File Edit Selection View Go Run Terminal Exam Help

Fullscreen

02:10:54

EXPLORER

PROJECT

QP

images

Reference

index.html

To_Trainees

src

dsausingj...

progusingj...

Customer...

Digitalite...

MusicOrd...

Tester.java

REFERENCE

Question X Customer.java

• This array has one to one correspondence with **bitRateAvailable** array
• The initial values of **bitRateAvailable** are given below:

bitRateCost	{15.0, 10.0, 20.0}
--------------------	--------------------

Note:

- This array is supplied. Hence no need to code

validateBitRate():

- This method validates the **bitRate(int)** and returns the **index(Integer)** based on the following logic:
 - Check if **bitRate** is present as one of the elements in **bitRateAvailable** array
 - If it is present, return the corresponding **index** position of the **bitRate** in **bitRateAvailable** array
 - Otherwise return -1

Example: If **bitRate** is 256, then this method would return 2

calculateOrderPrice():

- This method calculates and sets the **orderPrice(double)** and generates the **orderId(String)** based on the below

Activate Windows
NJA2YWN1ODQwM2VmNDMzZTY5ZjcwYzUwIEN0YXZhISwVVBRnhA= — 02:10:54
You're being proctored!

PROJECT BASED PROBLEMS

File Edit Selection View Go Run Terminal Exam Help

Fullscreen

EXPLORER Reference Question x Customer.java

PROJECT

QP

images

Reference

index.html

To_Trainees

src

dsausingj...

progusingj...

Customer...

Digitalite...

MusicOrd...

Tester.java

calculateOrderPrice(): 02:10:51

- This method calculates and sets the `orderPrice(double)` and generates the `orderId(String)` based on the below logic:
- Invoke `validateCustomerDetails()` method of `Customer` class
- If the `musicFormat(String)` is available in the `musicFormatAvailable(String[])` array and the above method returns `true`, then
 - Invoke `validateBitRate()` method to get `bitRateIndex`.
 - If `bitRateIndex` is not `-1` and the `durationInSec(int)` is greater than `0`. If so, then:
 - Set the `baseCost` (`double`) with the corresponding `cost(double)` of `bitRate` in `bitRateCost(double[])` array by using above obtained `bitRateIndex`
 - Add currency `3` to the `baseCost` for each minute of music
Note: Perform **integer division**
 - Add `5%` service tax to the updated `baseCost`
 - Set `orderPrice(double)` with the above calculated `baseCost`
 - Invoke `generateOrderId()` method
 - Otherwise, set `orderPrice` to `-1.0` and `orderId` to `NA`
- Otherwise, set `orderPrice` to `-1.0` and `orderId` to `NA`

Note: Perform **case-insensitive** comparison

Example: If the `custName` is `Bobby`, `emailId` is `bob@xyz.com`, `age` is `19`, `itemName` is `Unforgiven`, `musicFormat` is `FLAC`, `bitRate` is `256` and `durationInSec` is `350` then, `orderPrice` would be currency `36.75` and `orderId` would be `c3RhdmFuLnBhdGls`

PROJECT BASED PROBLEMS

Fullscreen

02:10:48

c3RhdmFuLnBhdGls

File Edit Selection View Go Run Terminal Exam Help

EXPLORER

PROJECT
QP
> images
> Reference
index.html
To_Trainees
src
> dsausingj...
progusingj...
Customer...
Digitalte...
MusicOrd...
Tester.java

JAVA DEPENDENCIES
project_9d5f...

Reference Question x Customer.java

- Add 5% serviceTax to the updated baseCost
- Set **orderPrice(double)** with the above calculated **baseCost**
- Invoke **generateOrderId()** method
 - Otherwise, set **orderPrice** to -1.0 and **orderId** to ◊NA◊
 - Otherwise, set **orderPrice** to -1.0 and **orderId** to ◊NA◊

Note: Perform case-insensitive comparison

Example: If the **custName** is ◊Bobby◊, **emailId** is ◊bob@xyz.com◊, **age** is 19, **itemName** is ◊Unforgiven◊, **musicFormat** is ◊FLAC◊, **bitRate** is 256 and **durationInSec** is 350 then, **orderPrice** would be currency **36.75** and **orderId** would be ◊U1001◊, considering it to be the first order.

Question 2: Data Structures:

[5 Marks]

Problem Statement:

Description: Consider a non-empty **inIntStack** (int Stack) containing integers, and a non-empty **inIntQueue** (int Queue) containing integers.

Write a Java method that accepts the above **inIntStack** and **inIntQueue** as input parameters and returns an **outIntStack**.