**https://venkivijay.github.io/Java-Solutions-TCS-Xplore-Proctored-Assessment/#/**

**TravelAgencies**

Create a class TravelAgencies with below attributes:  
regNo – int  
agencyName – String  
pakageType – String  
price – int  
flightFacility – boolean

Write getters, setters for the above attributes . Create constructor which takes parameter in the above sequence.

Create class Solution with main method. Implement two static methods – findAgencyWithHighestPackagePrice and agencyDetailsforGivenIdAndType in Solution class.

**findAgencyWithHighestPackagePrice method:**

This method will take array of TravelAgencies objects as an input parameter and return the highest package price from the given array of objects.

**agencyDetailsForGivenldAndType method:**

This method will take three input parameters -array of TravelAgencies objects, int parameter regNo and String parameter packageType. The method will return the TravelAgencies object based on below conditions.

1. FlightFacility should be available.
2. The input parameters(regNo and packageType) should matched with the regNo and packageType of TravelAgencies object.

If any of the above conditions are not met, then the method should return null. Note : Same Travel agency can have more than one package type. Travel agency and package type combination is unique. All the searches should be case insensitive.

The above mentioned static methods should be called from the main method.

For **findAgencyWithHighestPackagePrice method** – The main method should print the highestPackagePrice as it is.

For **agencyDetailsForGivenldAndType method** -The main method should print the AgencyName and price of the returned object.The AgencyName and price should be concatinated with : while printing.

**Input**

123

A2Z Agency

Platinum

50000

true

345

SSS Agency

Gold

30000

false

987

Cox and Kings

Diamond

40000

true

888

Global Tours

Silver

20000

false

987

Diamond

**Output**

50000

Cox and Kings:40000

**NavalVessel**

Create a class NavalVessel with the below attributes:  
vesselld – int  
vesselName – String  
noOfVoyagesPlanned – int  
noOfVoyagesCompleted – int  
purpose – String  
classification – String

Write getters, setters for the above attributes. Create constructor which takes parameter in the above sequence except classification.

Create class Solution with main method. Implement two static methods – findAvgVoyagesByPct and findVesselByGrade in Solution class.

**findAvgVoyagesByPct method:**

This method will take two input parameters -array of NavalVessel objects and int parameter percentage. The method will return the average of the noOfVoyagesCompleted attribute from NavalVessel objects which meets the percentage requirement as mentioned below.The percentage for each navalVessel is calculated as percentage .(noOfVoyagesCompleted \* 100)/noOfVoyagesPlanned. If the above calculated percentage is greater than or equal to the percentage passed as the parameter, then those objects should participate in the average calculation. If there are no objects with the mentioned criteria, then the method should return 0.

**findVesselByGrade method:**

This method will take a String parameter purpose, along with the array of NavalVessel objects. The method will return the NavalVessel object, if the input String parameter matches with the purpose attribute of the NavalVessel object. Before returning the object, the classification should be derived as mentioned below. This classification value should be assigned to the object before returning. If any of the above conditions are not met, then the method should return null. The classification attribute should be calculated based on the percentage calculation mentioned above. If the percentage = 100 then classification should be ‘Star’, if percentage is between 99 and 80 then classification should be ‘Leader, if percentage is between 79 and 55 then classification should be ‘Inspirer otherwise the classification should be ‘Striver’ If the returned value is null then it should print “No Naval Vessel is available with the specified purpose”.

Before calling these static methods in main, use Scanner object to read the values of four NavalVessel objects referring attributes in the above mentioned attribute sequence (except the classification attribute). Next, read the value for percentage and purpose.

## Input

111

Rojer

100

100

Passenger

222

Kalam

200

100

Goods

333

Yashwin

400

300

Army

444

Thanishwini

500

500

Welfare

75

Army

## Output

300

Yashwin%Inspirer

**Autonomous Car**

Create a class AutonomousCar with the below attributes:  
carld – int  
brand – String

noOfTestsConducted – int  
noOfTestsPassed- int  
environment – String  
grade – String

Write getters, setters for the above attributes . Create constructor which takes parameter in the above sequence except grade.

Create class Solution with main method. Implement two static methods – findTestPassedByEnv and updateCarGrade in Solution class.

**findTestPassedByEnv method:**

This method will take two input parameters -array of AutonomousCar objects and string parameter environment. The method will return the sum of the noOfTestsPassed attribute from autonomousCar objects for the environment passed as parameter. If no autonomousCar with the given environment is present in the array of AutonomousCar objects, then the method should return 0.

**updateCarGrade method:**

This method will take a String parameter brand, along with the array of AutonomousCar objects. The method will return the autonomousCar object, if the input String parameter matches with the brand attribute of the autonomousCar object. Before returning the object, the grade should be derived based on the rating calculation mentioned below. This grade value should be assigned to the object. If any of the above conditions are not met, then the method should return null. The grade attribute should be calculated as follows: rating .(noOfTestsPassed \* 100)/noOfTestsConducted If the rating > = 80 then grade should be ‘A1’, otherwise the grade should be ‘B2’.

The above mentioned static methods should be called from the main method. For findTestPassedByEnv method – The main method should print the totalTestPassed as it is, if the returned value is greater than 0, or it should print “There are no tests passed in this particular environment”.

For updateCarGrade method – The main method should print the brand and grade of the returned autonomousCar object. The brand and grade should be concatinated with :: while printing. eg:- Tesla::A1, where Tesla is the brand and Al is the grade.

If the returned value is null then it should print “No Car is available with the specified brand”. Before calling these static methods in main, use Scanner object to read the values of four autonomousCar objects referring attributes in the above mentioned attribute sequence (except grade attribute). Next, read the value for environment and brand.

## Input

100

Tesla

1000

500

Hills

200

Ford

2000

1500

Desert

300

Royce

3000

1700

Hills

400

Mercedez

1000

400

Desert

Desert

Mercedez

## Output

1900

Mercedez::B2

**Movie Find Avg Budget By Director**

Create a class Movie with the below attributes: movieId - int director - String rating - int budget - int

Write getters, setters and parameterized constructor in the above mentioned attribute sequence as required.

Create class Solution with the main method.

**Implement two static methods**

findAvgBudgetByDirector

getMovieByRatingBudget in Solution class.

**findAvgBudgetByDirector method:**

This method will take two input parameters - array of Movie objects and string parameter director. The method will return the average of the budget attribute from Movie objects directed by the director passed as parameter. If no movie with the given director is present in the array of movie objects, then the method should return 0.

**getMovieByRatingBudget method:**

This method will take two int parameters rating and budget, along with the array of movie objects. The method will return the movie object, if the input parameters rating and budget, matches with the rating and budget attribute of the movie object respectively.

Also check if rating is a factor of budget (eg: 3 is a factor of 12 because 3 divides 12 without leaving a remainder). If any of the conditions are not met, then the method should return null.

Note : No two movie object would have the same value for rating and budget attributes. All the searches should be case insensitive. The budget mentioned are in crores and in INR.

These above mentioned static methods should be called from the main method.

**For findAvgBudgetByDirector method** –

The main method should print the average budget as it is if the returned value is greater than 0, or it should print "Sorry - The given director has not yet directed any movie".

**For getMovieByRatingBudget method** –

The main method should print the movieId of the returned movie object. If the returned value is null then it should print "Sorry - No movie is available with the specified rating and budget requirement".

Before calling these static methods in main, use Scanner object to read the values of four Movie objects referring attributes in the above mentioned attribute sequence. Next, read the value for director, rating and budget.

## Input

1101

GVM

4

100

1201

Shankar

5

500

1301

Shankar

3

50

1401

GVM

5

300

GVM

5

300

## Output

200

1401

**Player Get Player Based On Level**

Create a class Player with below attributes:

playerId - int

skill - String

level - String

points - int

Write getters, setters and parameterized constructor in the above mentioned attribute sequence as required.

Create class Solution with the main method.

Implement two static methods - findPointsForGivenSkill and getPlayerBasedOnLevel in Solution class.

**findPointsForGivenSkill method:** This method will take two input parameters - array of Player objects and string parameter skill. The method will return the sum of the points attribute from player objects for the skill passed as parameter. If no player with the given skill is present in the array of player objects, then the method should return 0.

**getPlayerBasedOnLevel method:** This method will take two String parameters level and skill, along with the array of Player objects. The method will return the player object, if the input String parameters matches with the level and skill attribute of the player object and its point attribute is greater than or equal to 20. If any of the conditions are not met, then the method should return null.

*Note :* No two player object would have the same value for skill and level attribute. All player object would have the points greater than 0. All the searches should be case insensitive.

These above mentioned static methods should be called from the main method.

**For findPointsForGivenSkill method** - The main method should print the points as it is if the returned price is greater than 0, or it should print "The given Skill is not available".

**For getPlayerBasedOnLevel method** - The main method should print the playerId of the returned player object. If the returned value is null then it should print "No player is available with specified level, skill and eligibility points".

Before calling these static methods in main, use Scanner object to read the values of four Player objects referring attributes in the above mentioned attribute sequence. Next, read the value for skill and level. Please consider the skill value read above as skill parameter for both the static methods.

## Input

101

Cricket

Basic

20

102

Cricket

Intermediate

25

111

Football

Intermediate

50

113

BaseBall

Advanced

100

Cricket

Intermediate

## Output

45

102

**Institution**

Create a class Institution with below attributes:

1. institutionId - int
2. institutionName - String
3. noOfStudentsPlaced - String
4. noOfStudentsCleared- int
5. location - String
6. grade - String

Write getters, setters for the above attributes. Create constructor which takes parameter in the above sequence except grade.

Create class Solution with main method. Implement two static methods - findNumClearancedByLoc and updateInstitutionGrade in Solution class.

**findNumClearancedByLoc method:** This method will take two input parameters - array of Institution objects and string parameter location. The method will return the sum of the noOfStudentsCleared attribute from institution objects for the location passed as parameter. If no institution with the given location is present in the array of institution objects, then the method should return 0.

**updateInstitutionGrade method:** This method will take a String parameter institutionName, along with the array of Institution objects. The method will return the institution object, if the input String parameter matches with the institutionName attribute of the institution object. Before returning the object, the grade should be arrived based on the rating calculation mentioned below. This grade value should be assigned to the object.If any of the above conditions are not met, then the method should return null.

**The grade attribute should be calculated as follows:** rating=(noOfStudentsPlaced \* 100)/noOfStudentsCleared If the rating &gt= 80 , then grade should be 'A'. Else, then grade should be 'B'

**Note:** No institution object would have the same value for institutionName attribute. All institution object would have the noOfStudentsPlaced value lesser than noOfStudentsCleared value. All the searches should be case insensitive.

The above mentioned static methods should be called from the main method.

For findNumClearancedByLoc method - The main method should print the noOfClearance as it is, if the returned value is greater than 0, or it should print "There are no cleared students in this particular location".

For updateInstitutionGrade method - The main method should print the institutionName and grade of the returned Institution object. The instituationName and grade should be concatinated with :: while printing. eg:- TCS::A, where TCS is the institution name and A is the grade. If the returned value is nullthen it should print "No Institute is available with the specified name".

Before calling these static methods in main, use Scanner object to read the values of four Phone objects referring attributes in the above mentioned attribute sequence (except grade attribute). Next, read the value for location and institutionName.

## Input

111

Amrita

5000

10000

Chennai

222

Karunya

16000

20000

Coimbatore

333

AppleTech

10000

12000

Chennai

444

Aruna

6000

10000

Vellore

Chennai

Karunya

## Output

22000

Karunya::A

[**Device Management**](https://github.com/venkivijay/Java-Solutions-TCS-Xplore-Proctored-Assessment/tree/master/Device_Management)

Create a class Phone with below attributes: phoneId - int os - String brand - String price - int

Write getters, setters and parameterized constructor in the above mentioned attribute sequence as required.

Create class Solution with main method.

Implement two static methods - findPriceForGivenBrand and getPhoneIdBasedOnOs in Solution class.

findPriceForGivenBrand method: This method will take two input parameters - array of Phone objects and string parameter brand. The method will return the sum of the price attribute from phone objects for the brand passed as parameter. If no phones with the given brand is present in the array of phone objects, then the method should return 0.

getPhoneIdBasedOnOs method: This method will take a String parameter os, along with the array of Phone objects. The method will return the phone object, if the input String parameter matches with the os attribute of the phone object and its price attribute is greater than or equal to 50000. If any of the conditions are not met, then the method should return null.

Note : No phone object would have the same value for os attribute. All phone object would have the price greater than 0. All the searches should be case insensitive.

These above mentioned static methods should be called from the main method.

For findPriceForGivenBrand method - The main method should print the price as it is if the returned price is greater than 0, or it should print "The given Brand is not available".

For getPhoneIdBasedOnOs method - The main method should print the phoneId of the returned phone object. If the returned value is null then it should print "No phones are available with specified os and price range".

Before calling these static methods in main, use Scanner object to read the values of four Phone objects referring attributes in the above mentioned attribute sequence. Next, read the value for brand and os.

## Input

111

iOS

Apple

30000

222

android

Samsung

50000

333

Symbian

HTC

12000

444

Paranoid

HTC

89000

Blackberry

aNdRoid

## Output

The given Brand is not available

222