Reach World Tour Management

**Grade settings**: Maximum grade: 100  
**Run**: Yes **Evaluate**: Yes  
**Automatic grade**: Yes

[***Click here to download the code skeleton***](https://cognizant.tekstac.com/pluginfile.php/69135/mod_vpl/intro/TourTrip%20Management.zip)

Reach World provides tour packages to various destinations. Due to the increased demand, they have approached Zee Software to automate their various requirements.

You are required to write Junit test case and check the correctness of the application developed.

**Functional Requirements:**

The application has the below classes and methods implemented.

You are provided with a model class TourPackage

**Component Specification: TourPackage** **(Model Class)**

|  |  |  |
| --- | --- | --- |
| **Type(Class)** | **Attributes** | **Methods** |
| TourPackage | String packageId  String source  String destination  int totalDays  String packageType  double perPersonCost | Necessary getters, setters are provided   Constructors are also provided |

Here, packageType can take a value either “Escorted” or “Independent” [Note: Values are case insensitive]

**Component Specification:**InvalidTourPackageException**(This class inherits the Exception Class)**

|  |  |
| --- | --- |
| **Type(Class)** | **Methods** |
| InvalidTourPackageException | Provided with a single argument constructor – InvalidTourPackageException(String message) |

The below are the requirements  implemented in the Utility class for which JUnit test cases are to be written and tested.

**Component Specification:**ServiceProvider **(Utility Class)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Type (Class)** | **Methods** | **Responsibilities** | **Exception** |
| Validate the package  type | ServiceProvider | public boolean validatePackageType (String packageType) | Validate the packageType.  If valid, return true , else this method should throw a user defined exception | Throw a user defined exception “InvalidTourPackageException”  if the packageType Is neither “Independent” nor “Escorted” |
| View TourPackage based on packageId | ServiceProvider | public TourPackage viewTourPackage(List<TourPackage> packageList,String packageId) | This method should return the TourPackage object with the packageId passed as parameter from list of TourPackage, which is also passed as parameter.  If the packageList is empty or if there is no tour package with the given packageId it should throw a user defined exception | Throw a user defined exception “InvalidTourPackageException” if the packageList is empty or if no TourPackage exists with the given packageId. |
| View the list of Tour Packages for a given packageType | ServiceProvider | public List<TourPackage> viewTourPackagesByPackageType (List<TourPackage> packageList ,String packageType) | This method takes the packageList and a packageType as an argument. It should return the list of TourPackage for the given packageType. If the packageList is empty it should throw a user defined exception. | Throw a user defined exception “InvalidTourPackageException” if the  packageList is empty |
| View all the TourPackages packageType wise | ServiceProvider | public Map<String,List<TourPackage> viewTourPackagesTypeWise(List<TourPackage> packageList) | This method takes the packageList as argument. It should return packageType wise, all the packages in the list. The return type is map, where the packageType is key and value is the List of TourPackage  belonging to that packageType.  If the packageList is empty it should throw a user defined exception. | Throw a user defined exception “InvalidTourPackageException” if the packageList is empty. |
| View the number of Tour Packages for each packageType based on the given packageCost | ServiceProvider | public  Map<String,Integer> countTotalTourPackagesForEachPackageType(List<TourPackage> packageList, double packageCost) | This method should return the number of Packages for each packageType in the packageList for the given packageCost (count of packages having perPersonCost less than or equal to the packageCost). It takes the packageList, packageCost  as arguments and returns a Map with key as packageType and value as count of tour packages whose perPersonCost is less than or equal to the package cost                If the packageList is empty it should throw a user defined exception. | Throw a user defined exception “InvalidTourPackageException” if the  packageList is empty. |

You need to write Junit test for the ServiceProvider class.

**Testing Scenarios:**

You are provided with a class  “ServiceProviderTest”  to do this testing.

**Note:**

To perform testing, the packageList should contain objects of Packages.

To do this, in the ServiceProviderTest class you are provided with a setup method.  Use this method to populate the static variable packageList in ServiceProviderTest class.  That is, create few objects for  TourPackage and populate the packageList given in ServiceProviderTest class with these objects and use that list to test the methods  in ServiceProvider class that needs a TourPackage list to be passed as attribute.

The below are the test methods to be implemented in ServiceProviderTest class.

|  |  |
| --- | --- |
| **Test Method** | **Scenarios / Responsibilities** |
| test11ValidatePackageTypeWhenIndependent | This method should test the validatePackageType method when “Independent” is passed as parameter |
| test12ValidatePackageTypeWhenEscorted | This method should test the validatePackageType method when “Escorted” is passed as parameter. |
| test13ValidatePackageTypeWhenInvalid | This method should test the validatePackageType method when invalid value is passed as parameter  validatePackageType is expected  to throw InvalidTourPackageException when type is invalid.  Write JUnit to test for the exception thrown  either by using appropriate annotation or by using try catch block. |
| test14ViewTourPackageForValidPackageId | This method should test the correctness of  viewTourPackage method for  an existing packageId.  Perform testing for the correctness of the value returned. |
| test15ViewTourPackageForInvalidPackageId | This method should test the correctness of  viewTourPackage method for a non-existing packageId.  Perform testing for the correctness of the value returned.  viewTourPackage method is expected  to throw InvalidTourPackageException when packageId does not exist.  Write JUnit to test for the exception thrown  either by using appropriate annotation or by using try catch block |
| test16ViewTourPackagesByPackageType | This method should test the correctness of  viewTourPackagesByPackageType method.  Perform testing for the correctness of the value returned. |
| test17ViewTourPackagesByPackageTypeForEmptyList | This method should test the correctness of viewTourPackagesByPackageType method for an empty packageList.  viewTourPackagesByPackageType method is expected  to throw InvalidTourPackageException when packageList is empty.  Write JUnit to test for the exception thrown  either by using appropriate annotation or by using try catch block |
| test18ViewTourPackagesTypeWise | This method should test the correctness of viewTourPackagesTypeWise method.  Perform testing for the correctness of the value returned. |
| test19ViewTourPackagesTypeWiseForEmptyList | This method should test the correctness of viewTourPackagesTypeWise method for an empty packageList.  viewTourPackagesTypeWise method is expected  to throw InvalidTourPackageException when packageList is empty.  Write JUnit to test for the exception thrown  either by using appropriate annotation or by using try catch block |
| test20CountTotalTourPackagesForEachPackageType | This method should test the correctness of countTotalTourPackagesForEachPackageType method.  Perform testing for the correctness of the value returned. |
| test21CountTotalPackagesForEachPackageTypeForEmptyList | This method should test the correctness of countTotalTourPackagesForEachPackageType method for an empty packageList.  countTotalTourPackagesForEachPackageType method is expected  to throw InvalidTourPackageException when packageList is empty.  Write JUnit to test for the exception thrown  either by using appropriate annotation or by using try catch block |

Implement the test methods and provide the needed annotation to all the methods in ServiceProviderTest class.

Also this class is provided with the annotation, so that the test methods are executed in ascending order of the test method names.

You are provided with a Main class with the main method to check the correctness of the test methods written in ServiceProviderTest class.

Having completed writing the test methods,  the code in Main class can be executed..