Elite Comfort Home

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

[**Click here to download the code template**](https://cognizant.tekstac.com/pluginfile.php/74069/mod_vpl/intro/ACMaintenance.zip)

Elite Comfort Home is a famous AC Maintenance company in the city. They have developed an application for taking various reports based on their AC maintenance service. The details of the various functions supported by the system are provided in this case study.

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 ServiceDetails

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

|  |  |  |
| --- | --- | --- |
| **Type(Class)** | **Attributes** | **Methods** |
| ServiceDetails | String serviceCode  String acType  String serviceType  String customerName  String phoneNumber  Date appointmentDate | Necessary getters and setters are provided.  Parameterized constructor and a no-argument constructor are provided as a part of the code skeleton. |

Here the acType can take a value either “Split” or “Window” or “Centralized” or “Portable”.

Here the serviceType can take a value either “Cleaning” or “Repair” or “Gas Refilling”

[Note: Values are case insensitive].

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

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

You are also provided with a utility class ServiceCenter and the below business requirements are implemented in it for which JUnit test cases are to be written and tested.

You are also provided with a utility class ServiceCenter with business methods.

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | **Type(Class)** | **Attributes** | **Methods** |
| ServiceCenter | ServiceCenter | List<ServiceDetails> serviceDetailsList | Getter and setter for the serviceDetailsList are provided. |

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

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Component Name** | **Type (Class)** | **Methods** | **Responsibilities** | **Exception** |
| Validating the AC type | ServiceCenter | public boolean validateAcType(String acType) | Validate the AC type.  If valid return true , else this method should throw a user defined exception | Throw a user defined exception “InvalidServiceDetailsException” if the AC type is not “Split” or “Window” or “Centralized” or “Portable”. |
| View Service Details by Service Code | ServiceCenter | public ServiceDetails viewServiceDetailsByCode(String serviceCode) | This method should return the Service Details object with the service code passed as parameter from serviceDetailsList.  If the serviceDetailsList  is empty or if there is no Service details with  the given serviceCode it should throw a user defined exception | Throw a user defined exception “InvalidServiceDetailsException” if the serviceDetailsList is empty or if there is no service details for the given service code. |
| View the list of Service details for a given service type | ServiceCenter | public List<ServiceDetails> viewServiceDetailsByServiceType(String serviceType) | This method takes the service type as argument. It should return the list of Service details for the given service type. If the serviceDetailsList is empty it should throw a user defined exception. | Throw a user defined exception “InvalidServiceDetailsException” if the serviceDetailsList is empty. |
| View Service details for each Appointment date | ServiceCenter | public Map<Date,List<ServiceDetails>> viewServiceDetailsAppointmentDateWise() | This method should return the Service details for each Appointment date from the serviceDetailsList. It returns a Map with key as Appointment date and value as Service details.  If the serviceDetailsList is empty it should throw a user defined exception. | Throw a user defined exception “InvalidServiceDetailsException” if the serviceDetailsList is empty. |
| View the count of Service details based on the AC type | ServiceCenter | public Map<String,Integer> countTotalCountForEachAcType() | This method should return the number of service details for each AC type based on the serviceDetailsList. It returns a Map with key as AC type and value as count of service details.  If the serviceDetailsList is empty it should throw a user defined exception. | Throw a user defined exception “InvalidServiceDetailsException” if the serviceDetailsList is empty. |

You need to write Junit test for the ServiceCenterclass.

**Testing Scenarios:**

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

**Note:**

To perform testing the serviceDetailsList should contain objects of ServiceDetails.

To do this, in ServiceCenterTest class you are provided with a setup method. Use this method to initialize the  serviceDetailsList  attribute in ServiceDetails class.

Create few objects for ServiceCenter, populate a list with these objects and set the serviceDetailsList to this list using the setServiceDetailsList method in ServiceDetails class.

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

|  |  |
| --- | --- |
| **Test Method** | **Scenarios / Responsibilities** |
| test11ValidateAcTypeWhenSplit | This method should test the validateAcType method when “Split” is passed as parameter |
| test12ValidateAcTypeWhenWindow | This method should test the validateAcType method when “Window” is passed as parameter |
| test13ValidateAcTypeWhenCentralized | This method should test the validateAcType method when “Centralized” is passed as parameter |
| test14ValidateAcTypeWhenPortable | This method should test the validateAcType method when “Portable”  is passed as parameter |
| test15ValidateAcTypeWhenInvalid | This method should test the validateAcType method when invalid value is passed as parameter  validateAcType is expected  to throw InvalidServiceDetailsException when AC type is invalid.  Write JUnit to test for the exception thrown either by using appropriate annotation or by using try catch block. |
| test16ViewServiceDetailsByCodeWhenValid | This method should test the correctness of viewServiceDetailsByCode method.  Perform testing for the correctness of the value returned. |
| test17ViewServiceDetailsByCodeWhenInvalid | This method should test the viewServiceDetailsByCode method when invalid value is passed as parameter  viewServiceDetailsByCode is expected  to throw InvalidServiceDetailsException when code is invalid.  Write JUnit to test for the exception thrown either by using appropriate annotation or by using try catch block. |
| test18ViewServiceDetailsByServiceType | This method should test the correctness of viewServiceDetailsByServiceType method.  Perform testing for the correctness of the value returned. |
| test19ViewServiceDetailsAppointmentDateWise | This method should test the correctness of viewServiceDetailsAppointmentDateWise method.  Perform testing for the correctness of the value returned. |
| test20CountTotalCountForEachAcType | This method should test the correctness of countTotalCountForEachAcType method.  Perform testing for the correctness of the value returned. |
| test21ViewServiceDetailsByServiceTypeForEmptyList | This method should test the correctness of viewServiceDetailsByServiceType method for an empty serviceDetailsList.  viewServiceDetailsByServiceType method is expected  to throw InvalidServiceDetailsException when serviceDetailsList is empty.  Write JUnit to test for the exception thrown  either by using appropriate annotation or by using try catch block |
| test22ViewServiceDetailsAppointmentDateWiseForEmptyList | This method should test the correctness of viewServiceDetailsAppointmentDateWise method for an empty serviceDetailsList.  viewServiceDetailsAppointmentDateWise method is expected  to throw InvalidServiceDetailsException when serviceDetailsList is empty.  Write JUnit to test for the exception thrown  either by using appropriate annotation or by using try catch block |
| test23CountTotalCountForEachAcTypeForEmptyList | This method should test the correctness of countTotalCountForEachAcType method for an empty serviceDetailsList.  countTotalCountForEachAcType method is expected  to throw InvalidServiceDetailsException when serviceDetailsList 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 ServiceCenterTest 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 ServiceCenterTest class.

Having completed writing the test methods, uncomment the code in Main class and execute the main method.