package com.cts.cdgallery.test;

import static org.junit.Assert.\*;

import java.util.ArrayList;

import java.util.List;

import java.util.\*;

import org.junit.rules.ExpectedException;

import org.junit.AfterClass;

import org.junit.BeforeClass;

import com.cts.cdgallery.model.CDInfo;

import com.cts.cdgallery.util.CDGallery;

import com.cts.cdgallery.exception.InvalidCDInfoException;;

import org.junit.Rule;

import org.junit.Test;

public class CDGalleryTest {

@Rule

public ExpectedException exceptionRule=ExpectedException.none();

private static List<CDInfo> cdList = new ArrayList<CDInfo>();

private static CDGallery cdGalleryObj=null;

static CDInfo c1;

static CDInfo c2;

static CDInfo c3;

private static List<CDInfo> cdInfoList=new ArrayList<>();

@BeforeClass

public static void setUp() throws Exception {

cdGalleryObj=new CDGallery();

//Create few CDInfo objects and add to cdList.

//Use that list to test all the methods in CDGallery class that requires a list of CDInfo

c1=new CDInfo("123","Avengers","marvel",2019,"Yes");

c2=new CDInfo("789","AP","MRproduction",2018,"No");

c3=new CDInfo("456","ginee","xyz",2001,"Yes");

cdInfoList.add(c1);

cdInfoList.add(c2);

cdInfoList.add(c3);

//cdGalleryObj.setCdList(cdInfoList);

}

@AfterClass

public static void tearDown() throws Exception {

}

// test the validateCDWorkingStatus method when a valid Status Yes is passed as parameter to this method.

@Test

public void test11ValidateCDWorkingStatusWhenYes() throws InvalidCDInfoException {

// Code here..

assertTrue(cdGalleryObj.validateCDWorkingStatus("Yes"));

}

//test the validateCDWorkingStatus method when a valid Status No is passed as parameter to this method.

@Test

public void test12ValidateCDWorkingStatusWhenNo() throws InvalidCDInfoException {

// Code here..

assertTrue(cdGalleryObj.validateCDWorkingStatus("No"));

}

//test the validateCDWorkingStatus method when an invalid Status is passed to this method.

@Test

public void test13ValidateCDWorkingStatusWhenInvalid() throws InvalidCDInfoException{

// Code here..

exceptionRule.expect(InvalidCDInfoException.class);

exceptionRule.expectMessage("Valid Working Status For CD are - Yes / No");

cdGalleryObj.validateCDWorkingStatus("abc");

}

//test the validateCDReleaseYear method when valid Year is provided

@Test

public void test14ValidateCDReleaseYearForValidYear() throws InvalidCDInfoException {

// Code here..

assertTrue(cdGalleryObj.validateCDReleaseYear(2001));

}

//test the validateCDReleaseYear method when invalid Year is provided

@Test

public void test15ValidateCDReleaseYearForInvalidYear() throws InvalidCDInfoException{

// Code here..

exceptionRule.expect(InvalidCDInfoException.class);

exceptionRule.expectMessage("Valid release year for CD is from 1999 till current year");

cdGalleryObj.validateCDReleaseYear(1990);

}

//test the countNoOfWorkingCDs method when a CD List is passed as parameter.

@Test

public void test16CountNoOfWorkingCDs() throws InvalidCDInfoException {

// Code here..

List<CDInfo> tmp=new ArrayList<>();

tmp.add(c1);

tmp.add(c3);

assertEquals(2,cdGalleryObj.countNoOfWorkingCDs(tmp));

}

//test the viewCDInfoBetweenReleaseYear method when from release date and to release date is passed as parameter exists in the cdList.

@Test

public void test17ViewCDInfoBetweenReleaseYear() throws InvalidCDInfoException{

// Code here..

List<CDInfo> temp=new ArrayList<>();

temp.add(c1);

temp.add(c2);

assertEquals(temp,cdGalleryObj.viewCDInfoBetweenReleaseYear(cdInfoList,2015,2020));

}

//test the countNoOfMoviesOfProducer method when a CD List is passed as parameter.

@Test

public void test18CountNoOfMoviesOfProducer() throws InvalidCDInfoException {

// Code here...

//Map<String,Integer> tempmap=

assertFalse(cdGalleryObj.countNoOfMoviesOfProducer(cdInfoList).isEmpty());

}

}