Mission 7.1 GradeDisplayer

D0745378 薛竣祐

GradeDisplayer的開發重點在正確的呼叫GradeCollecter跟GradeSorter，並提供相關的參數，並確定接收到的回傳結果正確的。測試時可以先以stub設計GradeCollecter跟GradeSorter回傳假結果，讓GradeDisplayer先以該假結果正常執行，並確認該做的事情無誤，例如印出成績等等。

GradeCollecter的開發重點在需要區呼叫GradeDB並將結果透過AverageComputer算出各科平均。可以mock來建立一個假的GradeDB，並將回傳結果寫死，傳進AverageComputer，可以先預設一個答案，並使用Assertion來驗證AverageComputer的結果是否正常。若AverageComputer還不可用，也可以透過一樣方式設一個假結果，並將對應的內容回傳至上層。

當GradeSorter和GradeCollecter完成時，GradeDisplayer可以將之前的stub設定部分註解掉，直接使用真實函式，並檢驗結果是否正常。

App

package GradeDisplayer;  
import java.util.Arrays;  
  
public class App {  
 public static void main(String[] args) {  
 App app = new App();  
 app.gradeDisplayer(1);  
 }  
 //main app  
 void gradeDisplayer(int classID){  
 int[] grade = gradeCollecter(classID);  
 grade = gradeSorter(grade);  
 System.*out*.println(Arrays.*toString*(grade));  
 }  
 //return grade[]  
 int[] gradeCollecter(int classID){  
 int[][] subjectGrade = gradeDB(classID);  
 return averageComputer(subjectGrade);  
 }  
 //return grade[]  
 int[] averageComputer(int[][] subjectGrade){  
 int peopleCount = subjectGrade.length;  
 int[] grade = new int[peopleCount];  
 for(int i=0;i<peopleCount;i++){  
 for(int j=0;j<subjectGrade[i].length;j++){  
 grade[i] += subjectGrade[i][j];  
 }  
 grade[i]/=subjectGrade[i].length;  
 }  
 return grade;  
 }  
  
 //return subjectGrade[][]  
 int[][] gradeDB(int classID){  
 if (classID==1){  
 return new int[][]{  
 {10,20,30,40},  
 {60,60,60,60},  
 {90,100,80,70}  
 };  
 }  
 if (classID==2){  
 return new int[][]{  
 {20,60,80,90},  
 {30,40,50,30},  
 {12,23,34,45}  
 };  
 }  
 return null;  
 }  
 //return a sorted grade[]  
 int[] gradeSorter(int[] grade){  
 for(int i=0;i<grade.length;i++){  
 for(int j=i+1;j<grade.length;j++){  
 if(grade[i]>grade[j]){  
 grade = dataSwaper(grade,i,j);  
 }  
 }  
 }  
 return grade;  
 }  
 //swap i,j int data[]  
 int[] dataSwaper(int[] data, int i, int j){  
 int tmp = data[i];  
 data[i] = data[j];  
 data[j] = tmp;  
 return data;  
 }  
}

AppTest

/\*  
 \* This Java source file was generated by the Gradle 'init' task.  
 \*/  
package GradeDisplayer;  
  
import org.junit.jupiter.api.BeforeEach;  
import org.junit.jupiter.api.Test;  
  
import java.util.Arrays;  
  
import static org.junit.jupiter.api.Assertions.\*;  
import static org.mockito.Mockito.\*;  
  
class AppTest {  
 App app;  
 @BeforeEach  
 public void setUp(){  
 app = *spy*(App.class);  
 }  
  
 @Test  
 public void averageComputerTest() {  
 //stub  
 *when*(app.gradeDB(10))  
 .thenReturn(new int[][]{{10,10,10},{20,20,20}});  
 int[][] subject = app.gradeDB(10);  
 int[] exp\_grade = app.averageComputer(subject);  
 int[] real\_grade = {10,20};  
 *assertArrayEquals*(exp\_grade, real\_grade);  
 }  
 @Test  
 void gradeSorterTest(){  
 int[][] useless = new int[][]{{1,2,3}};  
 *when*(app.averageComputer(useless))  
 .thenReturn(new int[]{30,20,10});  
 int[] grade = app.averageComputer(useless);  
 int[] exp\_grade = app.gradeSorter(grade);  
 int[] real\_grade = {10,20,30};  
 *assertArrayEquals*(exp\_grade,real\_grade);  
 }  
 @Test  
 void gradeCollecterTest(){  
 *when*(app.gradeDB(3))  
 .thenReturn(new int[][]{{10,20,30},{60,70,80},{100,90,80}});  
 int[] exp\_grade = app.gradeCollecter(3);  
 System.*out*.println(Arrays.*toString*(exp\_grade));  
 int[] real\_grade = {20,70,90};  
 *assertArrayEquals*(exp\_grade,real\_grade);  
 }  
  
}

過程：

先將func的各參數與回傳設定好，由最上層開始撰寫，先以spy方式預設下層的結果，並嘗試呼叫，確認印出結果無誤時，往下層移動，重複步驟，也是以spy預設下層結果，逐漸嘗試。直到最下層時，可以先寫一份假程式，例如DB可以直接回傳假資料，並回推上層是否能正確呼叫，或是實際完成該程式，像是Swap可以實作真實程式，並往上推，去掉stub後測試是否正常，並逐漸回到最上層，完成整體程式。