Class 1 – Arrays and Collections

Introduction

These exercises are about writing classes using arrays and collections. However, a further purpose is to make you reflect on which collections are appropriate to use, and the quality of your code. You can use the IntelliJ IDEA environment or if you prefer Eclipse or Netbeans IDE to write your code.

1. The MarksArray example

Copy the Marks project, marks.zip, from Moodle to your workspace. This is an IntelliJ project. If you are working with a different environment, you can import the source .java files to a new project you create in your development environment.

1) The MarksArray class is incomplete. It is intended that it should hold the students and marks for module moduleName given in year.
2) Complete the constructor MarksArray(String moduleName, int year, int numStudents) which should initialise the members of a MarksArray object. The length of students and marks arrays should be numStudents.
3) The MarksArray(String moduleName) constructor uses this to initialise the members of a MarksArray object with the DEFAULT xxx values. What are the benefits of using this?
4) Complete addStudent(String name, int number) which should add name in slot number of students only if that slot is not already occupied. It should return whether the operation was successful.
5) Complete the accessor method getStudent(int number) which returns the name in slot number of students.
6) Complete setMark(int number, int mark) which sets the mark for student number and getMark(int number) which returns the mark, in both cases if and only if there is a student with that number.
7) Complete list() which prints the names and marks of all the students taking the module, one per line. Don't print any empty slots.
8) Run the main method and check that your code works.
9) This is not a good implementation. Consider how this class might be used. What problems does this class have? Identify as many as you can.

2. Hacked!

1) A student taking the module has kindly offered to analyse the marks and provide some statistics. Is the code safe? Is there any way that he might corrupt it? Do all the fields and methods have the right visibility?
2) Add the following code at the end of the main method

```
float average = new Adversary().analyse(marks);
//Check that we have what we want
marks.list();
System.out.println("Average =" + average);
```

Run the main method again. What has happened to your marks? Why? How can you prevent this sort of exploit?

3. Collections

1) What would be a better choice of collection to use in this class? Why?
2) Rewrite the class to use this collection.