

COMP 182 Project 1: Vectors (15 points)

Due: 20:00 Friday February 16, 2007

Overview: You will be writing an application which could be used to keep track of the scores earned by students and calculate their class percentage. The requirements will be simpler than what would be necessary to keep the scores for this section of COMP182, but could be modified to work fairly easily. Your program will use vectors (dynamically allocated arrays). If you want to start this assignment prior to the time that vectors are covered in class replace the word vector with the word array and then modify your program after they are covered (the necessary changes will be minor - approximately 21 lines of code). For this project you are not allowed to use `java.util.Vector` (even if you know what it is). You have to implement the vectors using arrays for yourself.

Requirements: You will write 3 classes.

The first class will be called `MyStudent`. It will contain a `String` name and two vectors of ints (assignScores and testScores) and in addition to the constructor at least 3 methods (`addAssignScore(int sc)`, `addTestScore(int sc)`, and `toString()`). The `addAssignScore(int sc)` method add (insert at the end) `sc` to the vector `assignScores`. The `addTestScore(int sc)` method will add (insert at the end) `sc` int to the vector `testScores`. The `toString()` method will return a `String` consisting of the name of the `MyStudent` followed by the assignment scores (seperated by tabs) followed by the test scores (seperated by tabs) followed by the overall percentage earned by the student to this point. The percentage should be calculated by computing average of the percentage earned on the assignments and the percentage earned on the tests (assume that every assignment is worth 10 points and every test is worth 20 points). If there are 0 assignments and/or 0 tests then the percentage earned on the assignments and/or tests should be treated as 0 (rather than using 0/0 and having your program crash). I suspect that this class will be approximately 100 lines (30ish of which will be comments).

The second class will be called `MyCourse`. It will contain a `String` `courseName` and a vector of `MyStudents` called `allStudents` and in addition to the constructor have at least 2 methods (`addStudent(Student st)` and `toString()`). The `addStudent(Student st)` method will add (insert at the end) `st` to the vector. The `toString()` method will return a `String` which starts with the name of the class and on each subsequent line contains the information for one `MyStudent` (hint: have this `toString()` use the `toString()` from `MyStudent`). I would suggest having a methods called `addAssignAll()` and `addTestAll()` which call the `addAssign(int sc)` and `addTest(int sc)` (from `MyStudent`) for each entry in `allStudents`. I suspect that this class will be approximately 100 lines (30ish of which will be comments).

The third class will be called `MyDriver1`. It will instantiate a single `MyCourse` and then allow interaction between the user and the program. See the sample interaction below for additional details. With the exception of using **your** name rather than **Noga**. I suspect that this class will require approximately 150-200 lines of code (40ish of which will be comments). If you plan out this class in detail before coding you will drastically reduce both the length of the code and the amount of debugging.

Submission: When you finish (prior to the deadline), log in to WebCT, go into the Assign Submits area, and upload/submit only the files `MyStudent.java`, `MyCourse.java`, and `MyDriver1.java`. It

would be a very good idea to submit the project at least several hours before the deadline (you can resubmit later if necessary). I suggest that when you submit, you request a confirmation email from the server. If you later want to claim the server lost your submission, I will want to see this email.

Grading: This assignment will be graded based upon coding style (comments, indenting, structure, useful var names, etc), whether it compiles and runs properly, how closely your output matches the sample output, and whether you followed directions (don't submit class files, don't use `java.util.Vector`, don't change the names of the classes, etc).

Bored: If you find the project too easy and finish early then submit the project as described (don't add extra features to the submission). The projects will get more difficult, but if you're bored and looking for something useful to do in the meantime, make a copy of your project and add features like:

1. reading an initial set of students/scores from a file,
2. writing the final set of students/scores to a file,
3. replacing your vectors with `java.util.Vector`,
4. adding quizzes/lab exams/midterms/final (% calculation changes)
5. add column headers
6. dropping students
7. modifying previously entered scores
8. fixing tabs (see end of the sample output for a annoying example)

Each individual feature is not too hard, but all together they should slow you down a bit.

Sample Interaction:

```
jnoga@nogahome:$ java MyDriver1
```

```
Welcome to Noga's Grade Recorder
```

```
Enter a command
```

```
> uuummm?
```

```
Not a valid command!!! Try help.
```

```
Enter a command
```

```
> help
```

```
Legal commands are:
```

```
student      (add a new student)
```

```
assignment   (add an assignment score for every student)
```

```
test         (add an test score for every student)
```

```
show         (displays the class)
```

```
help         (gives this screen)
```

```
quit         (ends the program)
```

```
Enter a command
```

```
> student
```

What is the student's name?

> Jose

Enter a command

> student

What is the student's name?

> Sara

Enter a command

> student

What is the student's name?

> Marla

Enter a command

> show

Noga's Class

Jose 0

Sara 0

Marla 0

Enter a command

> assignment

Jose's score?

> 8

Sara's score?

> 6

Marla's score?

> 10

Enter a command

> assignment

Jose's score?

> 8

Sara's score?

> 10

Marla's score?

> 8

Enter a command

> show

Noga's Class

Jose 8 8 40

Sara 6 10 40

Marla 10 8 45

Enter a command

> student

What is the student's name?

> Archibald

Enter a command

> test

Jose's score?

> 16

Sara's score?

> 20

Marla's score?

> 12

Archibald's score?

> 20

Enter a command

> show

Noga's Class

Jose	8	8	16	80
------	---	---	----	----

Sara	6	10	20	90
------	---	----	----	----

Marla	10	8	12	75
-------	----	---	----	----

Archibald	20		50	
-----------	----	--	----	--

Enter a command

> quit