# Julian Augley – 2107012A

Assumptions used in this program:

1. I assumed that we do not need to check whether the class list is empty or has more than one class when the ‘View Attendances’ button is clicked. I did not implement any error checking for this. In addition there is no error handling for trying to delete a class from an empty list, beyond the check for an existing id.
2. The assumptions stated in the assignment specification means I have not implemented any kind of input validation for the two input files, and I have also assumed this applies to the user inputs when adding a class. If the files or textfield inputs have even slight differences in format, specifically if the class id does not match the pattern ‘char char integer’ the program will not function as per specification and throw an exception. The program does check whether the id is longer than three characters in length and produce a warning, but does not check for id format. I could have implemented a check for the correct format using e.g.for (int i=0; I < 2; i++){ ‘if (charAt(i) >= 'a' && charAt(i) <='z') to test for alphabetic characters for the first two characters and a similar test for the third character, and proceed according to the outcome.
3. I also assumed references to averages meant specifically arithmetic mean, hence, averages have been calculated as: sum of elements/number of elements in sum. For example, overall mean attendance is the sum of mean attendances divided by the number of classes with a mean attendance, including where mean attendance is zero.

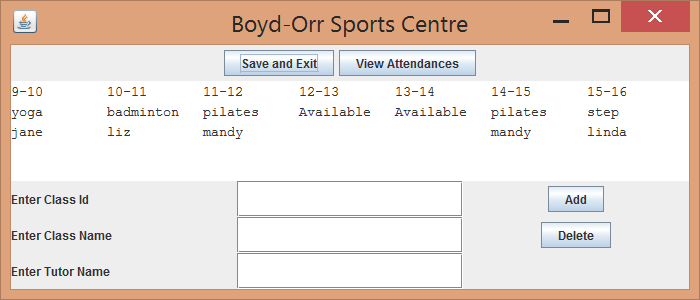
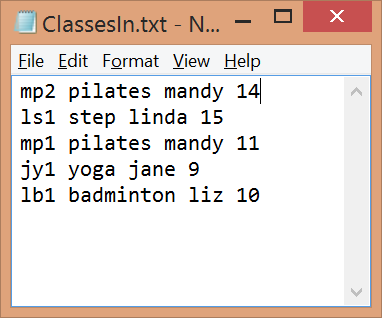
The program I have submitted meets specifications, subject to the assumptions stated here.

The test data used were the supplied ‘ClassesIn.txt’ and ‘AttendancesIn.txt’ files. Expected outputs were produced by the program (means etc were all calculated correctly)

The following screendumps demonstrate that correct functionality was implemented as per the specification sheet (subject to the assumption caveats stated at the start of this report).

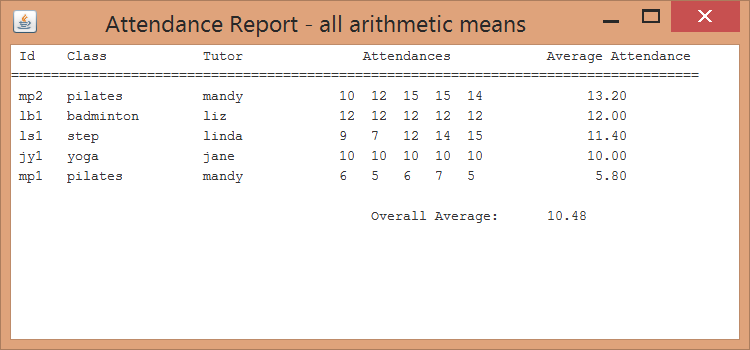
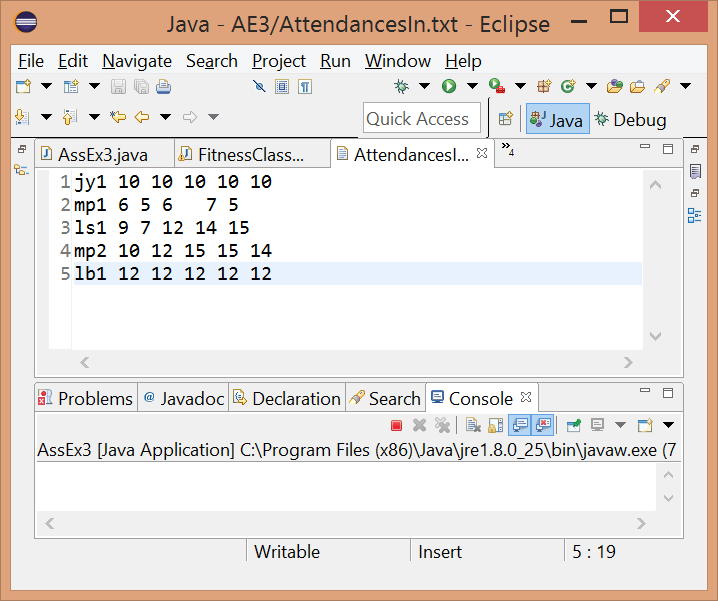
1. Functionality: ClassesIn file is read and information extracted correctly, timetable display is correctly formatted and displayed. Figure 1 contains screendumps of the ClassesIn.txt file and the GUI with a properly formatted timetable.

Figure 1: Screesnshots showing prgram has correctly extracted class data from ClassesIn.txt file, and properly formatted the timetable display on the GUI



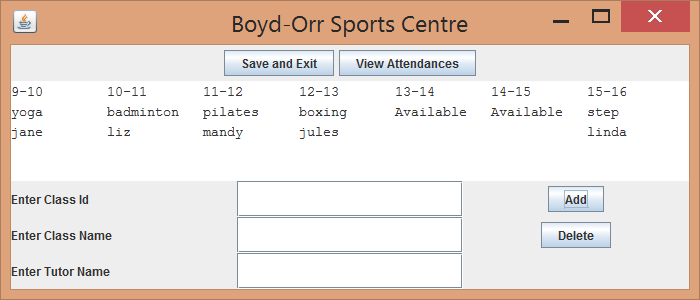
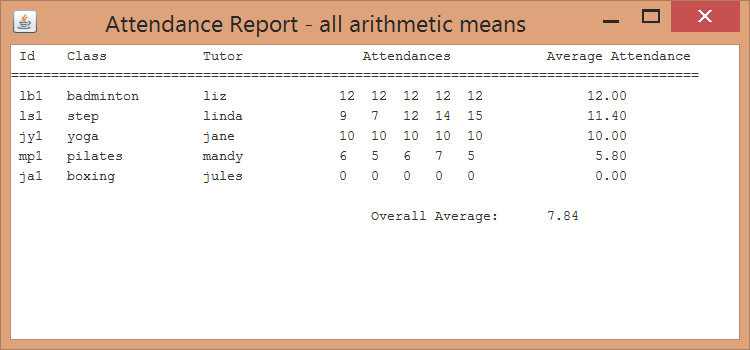
1. Functionality demonstrated: Attendance file is read in correctly, attendances stored correctly, report frame is properly formatted and averages are correctly calculated; figure 2 contains screendumps of Attendance Report window and attendances in file after one click of view attendances button.

Figure 2: Screenshots showing correct formatting of attendance report and that the program has correctly read in and stored the attendance file (attendancesin.txt)



1. Functionality: Closing attendance report window does not close the program, and delete and add buttons work correctly. Figure 3 shows screendumps of updated timetable display with a class deleted (mp2, pilates, mandy) and a class added (ja1, boxing, jules), and the attendance report with the updated classes and attendance means.

Figure 3: Screenshot demonstrating add and delete button functionality, correct recalculations of averages, and that closing the attendance report window does not close the program



1. Functionality: Save and exit button produces a file with the correct classes and formatting. Figure 4 shows screenshots of both the timetable display and the classes out file contents. In this case, the classes are listed in ascending time order (although this was not required, the coding just ended up with this functionality as classes were stored in the array according to time)

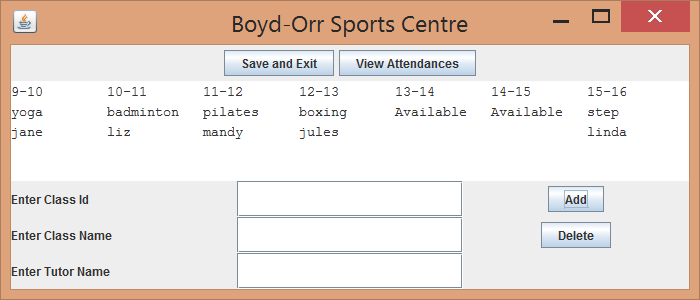
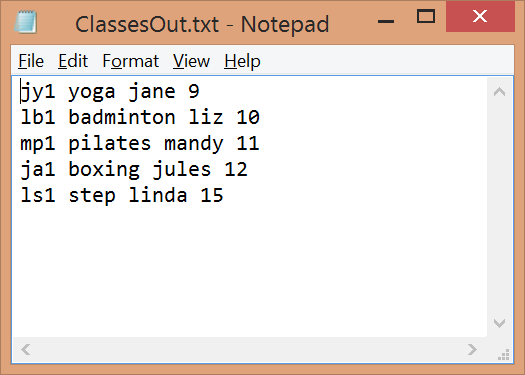


Figure 4: Screesnshot demonstrating correct formatting and production of classesoutfile, with the updated class data, and the timetable display

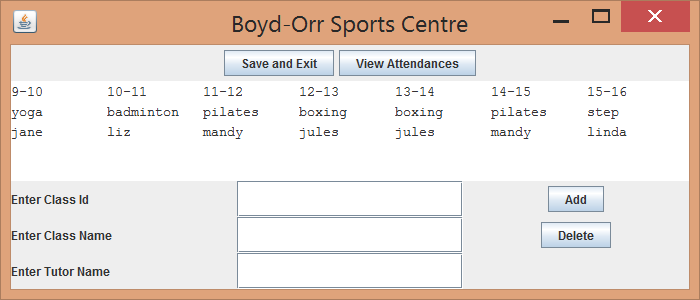
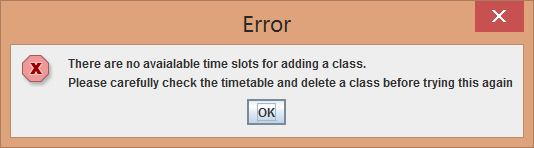
1. Functionality: Warning is produced when trying to delete a class that doesn’t exist. Figure 5 is a screendump of the JOptionPane produced in this scenario.



Figure : Warning when trying to delete a class that doesnt exist in the class list

1. Functionality: Warning is produced if trying to add a class to an already full list. Figure 6 contains screenshots showing JOptionPane and full timetable on GUI.

Figure 6: Screenshots demonstrating warning if trying to add a class to an already full list



1. Functionality: A warning is displayed if user tries to add a class with a duplicate id. Figure 7 is a screenshot of the JOptionPane warning produced.

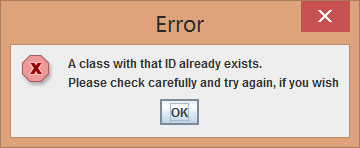


Figure : Warning message shown if trying to add a class with a duplicate id.

1. Functionality: A warning is shown when user tries to add a class but leaves one or more textfields empty. Figure 8 contains screenshots with the warning and the GUI state that produced the warning when the ‘Add’ button was clicked.

Figure 8: Warning message when textfields are empty when trying to add a class

