Introduction To Programming: Independent Investigative Effort 5

(See Canvas→Assignments for due dates and marks. Late submissions incur a 10% penalty on the full mark for each working date late; submission cut-off is 5 working days late. Marks turnaround time is approx. 10 working days after submissions close.)

Having trouble with watching recordings, usernames, passwords, access, etc.? Please call the RMIT IT Service and Support Centre for quick help on 03-9925 8888 and remember to ask for a reference number and pass it on to your instructor.

Need extensions or special consideration? Please apply for special consideration online.

Please follow/complete all steps below in the given sequence:

1. Please check your official RMIT Student email account for important course communication.

2.a. Watch any unwatched recordings of the **Weekly Live Lecture** and do all missed tutorials before going further. Gaps in programming concepts will lead to difficulties. If you need further help, please also watch your Group's chat recordings.

2.b. If you need help in addition to what has been shown in the compulsory weekly live lecture, you are also expected to speak to your **group tutor via discussion forums** and attend/watch their live sessions. Please note that group tutors cannot debug your assessment code on your behalf as debugging is a part of every programming assessment.

3. Check any available feedback of your previous submissions and if you have any unresolved questions or if you need further feedback, post the relevant parts of your submitted work in a new post under Canvas→Discussions→Independent Investigative Effort and ask from your instructor. E.g. you can ask "*Gayan showed _____ but I did mine like ______, so which is the better approach and why?*", etc. Please note that the university requires teaching to be conducted in an equitable manner so please only use email for matters such as special consideration.

4. How did you go during the past week? Please give feedback to Gayan so he can improve your learning experience, before it's too late, during this study period itself!

5. This week's programming task will cover some concepts required by Assignment 2 and 3. You should aim to get the help of your tutors and make further revisions.

Coding exercise steps (Hint: Need help? Ask your tutor via Canvas→Discussions→"IIE04"):
Follow Canvas→Modules→Week 5 first. If you did not complete your IIE04, please complete it now by following the 28/Sep/2020 live lecture (without doing so, you would be missing important steps and requirements of this IIE).

Make a copy of the IIE04 solution that uses GTerm (ensure GTerm is updated to 2020.08.25 or newer as shown during the 7/Sep weekly live lecture) and rename it to IIE05. Implement the rest of the steps in your .java file. You may need to rename the .java file to suit your application's behaviour before submission. Remove any irrelevant **justification comments** and new ones as required by the assignments (refer to assignment specifications). **Your code must not be about student records** to obtain full marks for this; user your imagination and adapt the program. Also note that, whenever the instructions use the term "array", it does not refer to the array obtained by the String class' .split method, unless otherwise stated.

a. Ensure that your program first asks the user how many records (numRecords) they would like to process. Then declare an array reference of an appropriate data type for each field of your records. E.g.

Instead of the variable:

```
String name;
```

Create an array to store many such values as:

```
String[] names;
```

If your code took inputs for 5 different pieces of information for each record, you must have 5 separate arrays. In other words, you will have an array reference for each column in the table but the arrays will be of different data types (whereas data in a table exist as Strings).

Now create the actual arrays based on the numRecords specified by the user for each of the arrays. For now, keep these two steps separate.

```
names=new String[numRecords];
```

Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.

b. When input String is split and the values are extracted/parsed and before they are added to the table, assign them to the arrays. i.e. if you have a names array and an ages array, names[0] and ages[0] must refer to the same student, names[1] and ages[1] should refer to the next student, etc. To achieve this, you will need to use your outer loop's loop variable to refer to the same index across all arrays. Of course, your code must not be about students. Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.

c. After 5b, your values would be stored in the arrays. Modify your addRowToTable to add these same values to the table as you did in IIE04. Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.

d. Investigative exercise: After taking all inputs are stored and the average of one of the numerical fields/columns is shown, find the lowest value stored in one of the arrays and display only that record in the table. E.g. you can find the lowest value stored in the GPAs array and show all information about that student (however, your code must not be about students). You may need to independently investigate in the GTerm documentation (Javadoc) for suitable methods in the 'method summary'. Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.

e. Optional to do but must follow when solution shown: What other interesting information/statistics can we find as a result of having the values stored in the arrays? Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.

f. Optional to do but must follow when solution shown: How can we make the program not ask for numRecords at the start and make it still work? Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.

f. Investigative exercise: Now modify the code (and remove relevant old code; there is no need to show the previous version separately) so that instead of using gt.println, the values are added to a table using GTerm. Independently investigate in the GTerm documentation (Javadoc) on how to add a table to the GTerm window and how to add a row to a table for each record. Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.

Submission Checklist for Step 5:
a. Ensure steps above have been followed in sequence.
b. Ensure that there are no red dots (compilation errors) in your code. Code with red dots are not valid Java and cannot be marked.
c. If you have not made a final submission for your Assignment 2, make a progress/dummy submission for Assignment 2 by submitting **your .java file to Canvas→Assignments→Assignment 2.** Do the same for Assignment 3 as well. Remember, you can overwrite this submission any time when you have a proper submission for your assignment.
d. Take screenshots of the code and the running program (as you did for IIE01) and **only embed screenshots** in a post **under Canvas→Discussions→Independent Investigative Exercise 5**. Please do not attach the images or post your answers in any other format as this would make the submission invalid. The mark for this week's work will be given based on this submission.
e. Ensure that your files are correct. If they are not, you can edit/delete your post and retry.

6. Optional: Make another copy of the program from #5 to perform the same via the console. All inputs must be taken via Scanner's .nextLine method and outputs must be via System.out.print or System.out.println methods. Note: This approach must not be used in major Assignments. Embed screenshots of this version of the program as a reply to your original post. **Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.**