Introduction To Programming: Independent Investigative Effort 6

(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→"IIE05"):
Follow Canvas→Modules→Week 6 first. If you did not complete your IIE05, please complete it now by following the 5/Oct/2020 live lecture (without doing so, you would be missing important steps and requirements of this IIE).

Your IIE05-5f solution as developed during 5/Oct/2020 is given at the end of this document for ease of reference. It takes the following overall layout:

```java
public class PleaseRenameMe {
        public static void main(String[] args) {
                GTerm gt = new GTerm(600, 400);
                // Rest of code
        }
}
```

a. Make a copy of the IIE05 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 IIE06 and move your code to fit the following structure:

```java
// The instructions in the comments must be followed when for all programs in Intro To Programming assessments from now on.
public class PleaseRenameMe {
        // The following are "object member variable" declarations.
        // All declarations at this level must be explicitly private and must not mention 'static'.
        // There should be no = (equals) signs here here.
        private GTerm gt;

        // The following is the "constructor" method
        public PleaseRenameMe() {
                // Object member variable declarations must be initialised before doing anything else.
                // Any reference to an object member variable must always start with "this."
                this.gt = new GTerm(600, 400);

                // Rest of code (remember to start with "this." when referring to object member variables)
        }

        public static void main(String[] args) {
                PleaseRenameMe prmObj=new PleaseRenameMe();
        }
}
```

Run the code and test it. It should look as it did in IIE05. i.e. there should be no difference to the end-user. Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.

b. Move all of your array declarations (you need to include all of them) and variable declarations that correspond with 'currentNumStudents' and 'maxNumStudents' (refer to IIE05-5f solution below) to the object member variable level (ensure that they are named to suit your program). Also ensure that they follow the guidelines given in the 5a's comments. The array creation and variable initialisations should still be inside the constructor. Ensure that, when a method refers to an object member variable (or array), it starts with `this.` ("this dot") at all times. Run the program and ensure that it works. Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.

c. Create the following method (it must be at the same level as the other methods, not inside any other methods):

```
public void refreshTable() {
        // 1. Clear the rows of the table.
        // 2. Loop through the arrays and add each record to the table
}
```

Implement the steps in the above method's comments. You will need to make the loop (step 2) repeat from 0 to currentNumStudents. Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.

d. Next, modify your old code (which should still be in the constructor) so that it does not invoke/use the .addRowToTable method. Instead, it calls the refreshTable method when it needs to display values in the table. Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.

e. After the statement that adds the table, add the following statement:

```
this.gt.addButton("Add", this, "addRecord");
```

Run the program and verify that there is a button named "Add".

Now implement the following method (it must be at the same level as the other methods, not inside any other methods):

```
public void addRecord() {
        // 1. Take comma separated inputs of a record
        // 2. Split the input
        // 3. Add the split input in to the array, increment currentNumRecords
        // 4. Call refreshTable
}
```

After implementing the above (and its comments), you may want to remove the loop-based record adding code from the IIE05 solution. i.e. now, when the user wants to add one record, they will select the "add" button. Are you stuck? Please ask your friendly tutor by creating a post in the relevant IIE forum.

f. Investigative exercise: With the help of your tutor, investigate in the GTerm documentation (Javadoc) for a suitable method to add a text field (or several). Modify your addRecord code so that it takes what is in the text field, instead of using .getInputString. 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 6**. 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.**

Simplified IIE05 5f solution from 5/Oct/2020 live lecture (part 2). Please refer to recording for full explanation:

```
public class StudentManager {
	public static void main(String[] args) {
		GTerm gt = new GTerm(600, 400);
		gt.setFontSize(16);

		int maxNumStudents = 1; // Integer.parseInt(gt.getInputString("Please enter number of students to
		                                                                 // process"));
		// Declarations
		String[] firstNames;

		// Creations (allocates memory)
		firstNames = new String[maxNumStudents];

		gt.addTable(500, 300, "First name\tLast name\tInitial\tYear\tGPA\tUnion?");

		int currentNumStudents = 0;
		String rawInput = gt.getInputString(
				"Student " + (currentNumStudents + 1) + "\nEnter first name,last name,middle initial,year joined,GPA,is union?");
		while (rawInput != null) {

			String[] fieldsOfStudent = rawInput.split(",");

			String firstName = fieldsOfStudent[0];
			while (firstName.isBlank())
				firstName = gt.getInputString("Must enter something for first name");

			// If we have reached the maximum capacity of the array
			if (currentNumStudents >= maxNumStudents) {
				// Change numStudents to what we want
				// Note: It does not change the original array's length
				maxNumStudents *= 2;
				gt.showMessageDialog("Expanding to " + maxNumStudents);
				String[] longerFirstNames = new String[maxNumStudents];
				// String[] longerLastNames...
				// double[] longerGPAs...

				int j = 0;
				while (j < currentNumStudents) { // Alternatively while(j<firstNames.length)
					longerFirstNames[j] = firstNames[j];
					// longerLastNames..
					// longerGPAs...

					j += 1;
				}
				firstNames = longerFirstNames;
				// lastNames=longerLastNames;
				// GPAs=longerGPAs;
			}

			firstNames[currentNumStudents] = firstName;

			String message = firstNames[currentNumStudents];
			gt.addRowToTable(0, message);

			currentNumStudents += 1;
			rawInput = gt.getInputString(
					"Student " + (currentNumStudents + 1) + "\nEnter first name,last name,middle initial,year joined,GPA,is union?");
		}

		// ----------------------------------

	}
}
```