uOttawa

SEG2105 - Introduction to Software Engineering - Fall 2018

Laboratory 2 (handed in as Assignment 1) – Object Oriented Concepts (4%)

- **Lab date:** Week of Sept. 17
- **Assignment due: Sep. 30 by 11h59pm**.
- **Group work and partners**: In this lab, and all other labs that involve programming, you will work in **groups of two.** If you have not previously arranged to work with someone in your lab section, then you will have to find someone immediately. If there are an odd number of students, the TA may permit an individual to work alone for now. No groups of 3 will be permitted.
- **Only one submission per group. Submit your work for parts 1 and 2 in a zip file (XXXXX_YYYYY.zip, where XXXXX and YYYYYY are your student ids).**
- **Seating**: In the early labs, each group will use one computer, although you will need an extra computer when you are doing later labs involving client-server work.
- **Saving work and accounts:** Your TA will give you any needed help regarding setting up your account. Before you leave the lab each day, make sure you have saved your work.
- **Using Eclipse:** You will be using Eclipse in this lab. Instructions on how to get started in the lab can be found here.
- **Exercises to do:** Work on the following exercises from the textbook. You must hand in only one copy of answers per group. Make sure the names and student numbers of both partners are clearly indicated. You must work on these questions during the lab and then finish off the work on your own time.

Analysis of various design alternatives

POINTCP

1. Before coming to the lab, you are asked to read and understand Section 2.9 of the textbook, starting on page 57. If you have not done this, take five minutes now to do it.
2. To start this lab, download the code for the PointCP example available at http://www.site.uottawa.ca/school/research/lloseng/supportMaterial/source/. Compile the code and run it.
3. By modifying the original version, implement **design 2, 3 and design 6**. When you hand in the lab, you will hand in the code you have written.

4. Modify the PointCPTest class to allow you to test the designs you have developed. Do a thorough series of tests to ensure your classes and interface work properly. You will hand in the code, and also a printout of the output generated by your tests.
5. Hand in your answers to exercises E26 (table of what you think will be the advantages and disadvantages of designs 2, 3 and 6), and E28-E30 (performance analysis, comparing all **three** designs you have implemented instead of Design 1 with Design 5 as the book says). To do this evaluation, for each design create random instances and then call each method many thousands of times, and then find the elapsed time in milliseconds for the fixed number of iterations. Make sure that your program runs each time for about 10 seconds, so you get a good measure of performance. Test each method separately. Run each version several times to ensure that your results are consistent and use the median result as your definitive result, plus give the maximum and minimum.

Design	**How cartesian coordinates are computed**	**How polar coordinates are computed**
Design 2: Store polar coordinates only	Computed on demand, but not stored	Simply returned
Design 3: Store cartesian coordinates only	Simply returned	Computed on demand, but not stored
Design 6: Interface with designs 2 and 3 as classes implementing it.	Depends on the concrete class used	Depends on the concrete class used

6. Hand in a description of how you did the tests, sample outputs from running the tests, the table and a discussion of the results.

ARRAYS

7. Compare the performance of **ArrayList**, **Vector** and ordinary **arrays**. You should do a series of experiments where you do each of the following tests with the three types of collection, timing the execution of each run. You should run each case several times on the same computer to obtain stable average timings.

   (a) Construct very large collections by putting random integers into each collection one at a time. The random integers should range in value from zero to nine. You should make each collection large enough so that the run takes at least 10 seconds to add the integers in the case of an ArrayList. You will have to do some initial experiments to find out what is a good size. You would use the same size of collection for ArrayList, Vector and the array. The ArrayList and Vector can be created by successively adding items and allowing them to grow, while the array has to be created at its full size and then populated with its contents. You could also try to experiment with the case where you do create the ArrayList and Vector initially with their full size.

(b) Construct very large collections as in (a). Then use iterators to sum the elements. Subtract the construction time to get a measure of how much time the iteration takes. Use a for loop for the array, and an Iterator for the Vector and ArrayList.

Present your data in suitable tables and draw conclusions from an analysis of the data. From your conclusions, develop recommendations to designers.

You have questions related to the assignment?

Please use the Piazza forums!