Department of Computer Science and Engineering
Dr B R Ambedkar National Institute of Technology (Deemed University)
Jalandhar -144022, Punjab, India

Case-Based Learning Exercise – My Baby Girl's Data

Course Title (and Type): Advanced Programming Concepts in Java (Elective)
Course Code: CSX-331
Engineering Programme and Semester: BTech in Computer Science and Engineering, 5th semester
Course Instructor: Paramvir Singh
Case Topic: Data Structures and Java Collections
Case Author: Paramvir Singh
Learning Mode: Case-Based Learning
Content-Type: In-Class/Home Assignment
Weightage: 6 out of 10 end-semester class assessment marks
Dates of Assignment, Expected Submission and Case Discussion: [TBD]

Exercise Objectives

- To learn the application of basic data structures like lists, stacks, queues, etc. in a real-world problem scenario.
- To conduct a rigorous team-based evaluation of various data structures applied and analyzed for the given case, on the basis of time, space and program code complexity.
- To exercise the Java Collections Framework through a treasure-hunt like solution search approach.
- To compare the strengths and weaknesses of various Java features, libraries and constructs in supporting data structures with other programming languages.

Case Description

My daughter Gunreet was born on March 24, 2012, and with her was born a new trail of data that has kept on growing since then. I maintained my assistance in bringing home a variety of items for her including toys, pencils, notebooks, charts, clothes, chocolates, etc., and, wait! I forgot nappies. And to make it worse, each one of them had a plethora of details, which I dint see but she did. These details include color, size, shape, characters, texture, popularity (in her little friend circle), and so on. Out of curiosity, one fine morning as she left for school I decided to compile the lists of them all. The plan was to handle the data in the best possible way in terms of the effective utilization of available time and space. I found it appropriate to map the physical space of her almirah to a part of my PC's memory space. I aimed to implement all sorts of operational possibilities, such as addition, deletion, update, search/locate, that those lists require in the form of a program, and leverage the storage information in taking important decision on which new items to buy, items to condemn, items to search quickly, etc. As much as possible, I wanted to match the arrangements in the physical almirah space with my PC's allocated memory space for storing all the lists.

Also, I wanted my lists and the underlying data to be handled with the best performance and in a platform-independent manner. I found that at a particular time, some of these lists might be quite small, while the others could be substantially large. I assume that there are lists for which I simply cannot predict the maximum size, and which would grow quickly at runtime putting a lot of extra burden on my space (e.g. managing her almirah space) management. For example, I could never predict how many candies would she bring from her school canteen and ask me to store them safely. At the same time, I am still undecided on how many candies can she consume in one go. A number of items in the same list might be identical, for instance, pencils. How should I identify them? …may

be using some identifiers. There are lists which simply cannot have identical items, for instance her course books. There are many item categories which are required to be accessed in the order of their purchase date, e.g. her eatables, medicines, etc. There are items with special constraints as well. For instance, I require a special permission from her mom to issue her another eraser until the previous one is fully used up. There are shared items as well. We often share her items like pencils, erasers, etc. My favorite among these commodities is her cardboard. But the most problematic item is her chair, which I often use to place my wifi modem, leading to a deadlock. Hence, appropriate scheduling techniques are required to handle such situations. Further, I want an access to all the data through a variety of devices including my laptop, smartphone, tablet, and my office PC as well.

Questions

1) Which data structures should I use to store and maintain those lists depending on the variety of features each item or list is expected to support?
2) Can java provide me a built-in support for all these lists and the underlying data structures through its well-known Collections Framework? How can I do it through Java?
3) Are the data structures implementations provided in the Java Collections Framework fool proof? Remember, I have very strict memory and time constraints.
4) Why shouldn't I look into other OO languages like C#, python, ruby, etc., where I might even find better implementations of these data structures?

Search Space or Resources

1) Google
2) Relevant websites
3) Books
4) Online tutorials
5) Developer blogs and forums
6) Research papers
7) Any other resources

Instructions

1) Read the case description carefully.
2) Discuss various case aspects with your group members.
3) Read the case questions carefully.
4) Discuss within the group about how to go about finding solutions to the questions.
5) Assign one question to each group member for which she will take full responsibility.
6) Make a one-week plan on how to search for possible answers to the questions.
7) Search through a variety of text, web or other possible sources.
8) Hold short daily group meetings to discuss the progress.
9) Evaluate all possible solutions for each answer in group meetings.
10) Prepare a short report on your solutions highlighting your approach toward finding them.
11) Upload your report to the shared *GitHub Education Repository.*
12) Formally present you solutions in the final Case Discussion sessions.
13) Contact the TAs or course instructor though *email*, *GitHub Issue Tracker* or *in-person* for any queries.

Additional Information: www.seabed.in

Signature of Course Coordinator