Discuss at least **two** of the following topics on Lists, Stacks, Queues, and Priority Queues for your initial post. Provide a code example, where necessary, to elaborate your thoughts.

* Using foreach loops to traverse elements in a collection.
* The difference between ArrayList and LinkedList, advantages and disadvantages of each.
* Collections methods for sorting, searching, shuffling a list, finding the largest and smallest elements.
* Describe Vectors, ArrayList, and Stacks explaining the differences.

Java is very versatile, given its vast abilities. The first topic of discussion is using foreach loops to traverse elements in a collection. Using a foreach loop to traverse elements in a collection “encapsulates the iteration code itself” without programmers having to do it (Minh, 2024). A programmer only needs to “specify what-to-do in each iteration” (Minh, 2024).

Here is an example of a foreach loop traversing elements in a collection (Oracle, 2018):

**>// Returns the sum of the elements of a**>

int sum(**int[] a**) {

int result = 0;

for (**int i : a**)

result += i;

return result;

}

The second topic of focus for this discussion board is the difference between ArrayList and LinkedList, as well as the advantages and disadvantages of each. It may be tricky to remember the difference between an ArrayList and a LinkedList since they both are defined under the List interface, which extends from Collection. Something to remember with an ArrayList is that if the capacity is exceeded, a larger array is created, and all elements in the array are copied into the new array (Liang, 2019/2025, sect. 20.5.2).

If you plan on using insertion or deletion, then it is best to use a LinkedList since it can grow and shrink (Liang, 2019/2025, sect. 20.5.2). Unlike the LinkedList, an ArrayList does not automatically shrink, but it does grow automatically, like LinkedList (Liang, 2019/2025, sect. 20.5.2). An important disquisition between the two is that LinkedList is better for inserting and removing elements at the start of a list while ArrayList works better for other operations (Liang, 2019/2025, sect. 20.5.2). LinkedList allows all methods that ArrayList minus trimToSize() but ArrayList does not have the getFirst, getLast, addFirst, and addLast methods as LinkedList does (Liang, 2019/2025, sect. 20.5).

**References**

Liang, D. Y. (2025). *Introduction to Java Programming and Data Structures: comprehensive version*. Pearson. (Original work published 2019)

Minh, N. H. (2024, July 18). *5 Ways to Iterate Collections in Java*. Codejava.net. https://www.codejava.net/java-core/collections/the-4-methods-for-iterating-collections-in-java#forEach

Oracle. (2018). *The For-Each Loop*. Oracle.com. https://docs.oracle.com/javase/8/docs/technotes/guides/language/foreach.html

**Assignment Requirements and Grading:**

* 1. An initial post of approximately 250 words is due by **Thursday, 11:59 p.m., CST**.
  2. For the initial post to be considered substantive, it should be at least 250 words in length and fully cover the topics being presented. Single sentence definitions or responses will not be awarded points.
  3. Submit your post by clicking on the assignment link above, then Create Thread. You must create a thread in order to view your peers' posts. Tip: Create your post in a Word document and then copy and paste your work into the thread.
  4. A minimum of three (3) responses, to the original threads of other students, of 100-200 words each are due by **Sunday, 11:59 p.m., CST**.
  5. To view the rubric grading criteria, click on the following link: [Discussion Board Grading Rubric.](https://content.bellevue.edu/cst/csd/rubricdbv3.pdf)

Hey, Adrian! You did a nice job of explaining the differences between ArrayList and LinkedList. Understanding the differences between the two helps determine the best one to use within a program. The ArrayList does not shrink by itself but grows automatically by storing the data in a larger array. The LinkedList also offers all the same methods that the ArrayList can use except for the trimToSize() since it does this itself. Overall, I think the ArrayList is usually the better option, but it still depends on the project. Java Collections continues to be a helpful addition when coding since it allows us to sort, search, shuffle a list, and find the largest and smallest elements.

Hi, Colton! After reading your post for this module, I think you did an excellent job explaining how to use a foreach loop to traverse a collection and the differences between ArrayList and LinkedList. I really enjoy the ease of use that Foreach () provides. As you said, it is a much more concise way to iterate over collections. I like how you connected the foreach loop to the functional programming style. As we go through this class, we learn more about making code easier to read. More often than not, the ArrayList seems to be the way to go rather than LinkedList.

Hello, Samir! I enjoyed reading your thoughts on the difference between ArrayList and LinkedList and using foreach loops to traverse elements in a collection. I elaborated on the same topics for my discussion board. The ArrayList can automatically expand by creating a new array to hold the elements. The code example for the foreach loop is a great demonstration of this Java feature. I was very excited to see the foreach loop in action, especially when compared to a foreach method or using an anonymous inner class, as we saw in our textbook. I found it especially helpful when completing our module assignments this week.