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

* Benefits of generics.
* Use and definition of generic classes and interfaces.
* Use of generic methods.
* Raw generic types for allowing backward compatibility.

In Java, there are generics that can be utilized when programming. According to GeeksforGeeks (2025), generics in Java are parameterized types that allow objects to be a “parameter to methods, classes, and interfaces.” These entities are called “generic entities” (GeeksforGeeks, 2016). There are two types of Java Generics: generic method and generic classes. A generic method uses a parameter(s) that is “cited by actual type” and returns a value (GeeksforGeeks, 2016). Generic classes are similar to non-generic classes but “contain a type parameter section” (GeeksforGeeks, 2016). An important part to remember is that an instance of a genetic type only accepts reference types (GeeksforGeeks, 2016). There are some disadvantages to using generics, like them being overcomplex for beginners, creating “overhead during runtime,” not supporting primitive types, and having “limited reflection” (GeeksforGeeks, 2016). Even though there are disadvantages, it has many advantages that can make it much more fitting to tackle different tasks in Java. One of the benefits of generics is the ability to reuse the code with any type (GeeksforGeeks, 2016). Generics also create safer code since errors are shown during compile time (GeeksforGeeks, 2016). Generics eliminates the need for type casting since the compiler casts for us (GeeksforGeeks, 2016). Lastly, generics make code easier to read and maintain while offering algorithmic implementation (GeeksforGeeks, 2016). According to Oracle, n.d., the same conventions and techniques for creating a generic class can be applied to creating genetic interfaces (GeeksforGeeks, 2016).

This is an example of using multiple type parameters (Oracle, n.d.):

public interface Pair<K, V> {

public K getKey();

public V getValue();

}

public class OrderedPair<K, V> implements Pair<K, V> {

private K key;

private V value;

public OrderedPair(K key, V value) {

this.key = key;

this.value = value;

}

public K getKey() { return key; }

public V getValue() { return value; }

}

**References**

GeeksforGeeks. (2016, March 1). *Generics in Java*. GeeksforGeeks. https://www.geeksforgeeks.org/generics-in-java/

Oracle. (n.d.). *Generic Types (The JavaTM Tutorials > Learning the Java Language > Generics (Updated))*. Docs.oracle.com. https://docs.oracle.com/javase/tutorial/java/generics/types.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, Megan! I really enjoyed reading your thoughts on the benefits of generics and the use of generic methods. The code you included accurately elaborates on what you are explaining. I like how casting is cast aside when using generics since it simplifies the code. I definitely foresee myself utilizing this in future Java programs. I like how you mentioned that generics take types as parameters because this is the main difference between generic and non-generic classes. Have you tried utilizing wildcards in Java yet? I have not so far, but I am interested to hear other people’s experiences with it.

Hi, Alondra! After reading your post for this module, you did a nice job of explaining the benefits of generics and the definition and use of generic classes and interfaces. I really find that the benefit of code reusability within generics in Java is a crucial factor. Reusing code is something that will offer to be used when programming. Especially when the code is firmly and securely developed, it saves us time and money reusing code and only having to update certain elements. So, knowing that generic classes and methods are capabilities of varying data types is a major bonus.

Hello, Jessica! You are spot on when you say that generics are a powerful feature in Java. I really see how the benefits of using this feature when we are coding. When coding using generic classes or methods, I really enjoyed referencing the Oracle website for more examples, so I like how you used it for further information. It also helps give an accurate and trustworthy way of using it since not all sites may have reliable code. I love the versatility of generics since we are not limited to one parameter type. I imagine we will utilize generic classes and methods much more going forward.