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

* Checked vs. Unchecked Exceptions.
* Exceptions: try/catch/throw.
* Exceptions: finally – both with or without catch.
* printStackTrace method of class Exception.
* Exceptions Declared and Not Handled.
* Create, Write, and Delete Files.
* Classes: FileSystem, Paths, Files, and BasicFileAttributes.
* Classes: BufferedOutputStream and OutputStream.
* Classes: BufferedReader and InputStreamReader.

Some of Java’s features include checked and unchecked exceptions. According to Liang (2019), the only unchecked exceptions are “RuntimeException, Error, and their subclasses,” the rest are all checked exceptions (sect. 12.3). The unchecked exceptions do not have to be stated in the method, but other method exceptions must be declared in the header (Liang, 2019, sect. 12.4.1). To use an exception or multiple exceptions the keyword “throws” is used in the header method (Liang, 2019, sect. 12.4.1).

In Java, another feature is the file class. The file class allows information about files to be retrieved, such as if a file exists, the length, the name, and the absolute path. It also can create, write, and delete files. A Java file can be created by using the

createNewFile() method (W3Schools, 2020). This method also checks to see if the file already exists; if it does, it will not be created. Another way to create and write data is by using the PrintWriter class (Liang, 2019, sect. 12.11.1). A file can be deleted using the delete() method (Devireddy 2021).

Devireddy (2021) provides an example of how to create a file.

**import java.io.File; // Import the File class  
import java.io.IOException; // Import the IOException class to handle errorspublic class CreateFile {  
 public static void main(String[] args) {  
 try {  
 File myObj = new File("filename.txt");  
 if (myObj.createNewFile()) {  
 System.out.println("File created: " + myObj.getName());  
 } else {  
 System.out.println("File already exists.");  
 }  
 } catch (IOException e) {  
 System.out.println("An error occurred.");  
 e.printStackTrace();  
 }  
 }  
}**

Devireddy (2021) provides an example of how to write to a file.

**import java.io.FileWriter;**

**FileWriter myWriter = new FileWriter("filename.txt");  
myWriter.write("Files in Java might be tricky, but it is fun enough!");  
myWriter.close();**

**References**

Devireddy, P. (2021, March 3). *Java Files, Java File Handling, Java Create and Write to a file, Read a File, Delete a File*. Medium; OOPS oriented Java Programming Language. https://medium.com/oops-oriented-java-programming-language/java-files-java-file-handling-java-create-and-write-to-a-file-read-a-file-delete-a-file-28061205b9aa

Liang, Y. D. (2019). *Introduction to Java programming and data structures: comprehensive version*. Pearson. https://plus.pearson.com/home?utm\_source=ereader

W3Schools. (2020). *Java Create and Write To Files*. W3schools.com. https://www.w3schools.com/java/java\_files\_create.asp

**Assignment Requirements and Grading:**

* An initial post of approximately 250 words is due by **Thursday, 11:59 p.m., CST**.
* 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.
* 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**.
* To view the rubric grading criteria, click on the following link: [Discussion Board Grading Rubric](https://content.bellevue.edu/cst/csd/rubricdbv3.pdf).