**✅ Interview-Focused Differences**

1. **Why are some exceptions checked and others unchecked?**
   * **Checked exceptions** handle recoverable problems (e.g., file not found).
   * **Unchecked exceptions** result from programming mistakes (e.g., accessing null).
2. **Can we convert checked to unchecked exceptions?**
   * Yes! By wrapping a checked exception inside a RuntimeException:

public void readFile() {

try {

new FileReader("file.txt");

} catch (IOException e) {

throw new RuntimeException(e);

}

}

3: **✅ How do you Handle Exceptions in Java Streams ?**

In Java Streams, exceptions can occur during operations like map(), filter(), forEach(), and collect(). Since lambda expressions cannot throw **checked exceptions** directly, handling them requires specific techniques.

1. **Use try-catch for simple cases – For quick error recovery, embed try-catch inside lambda expressions.**
2. **Converting Checked Exceptions to Unchecked Exceptions in Java Streams**
3. **Create reusable methods – For complex logic, wrap exception-prone code in a separate method.**
4. **Use Optional – For optional outputs, handle exceptions in a fluent and clean way.**
5. **Use custom wrappers – For checked exceptions, define a custom functional interface.**
6. **Log exceptions – Always log critical exceptions to ensure proper debugging and analysis.**