**Exception Hendling**

1. If there are no pointers in Java, then why do we get NullPointerException?
2. Arrange the three classes String, StringBuffer and StringBuilder in
3. the order of efficiency for String processing operations?
4. Exception Handling
5. What is the base class for Error and Exception classes in Java?
6. What is Exception Handling in Java?
7. Can we re-throw an Exception in Java?
8. What is the concept of Exception Propagation?
9. When we override a method in a Child class, can we throw an
10. additional Exception that is not thrown by the Parent class method?
11. When is UnsupportedOperationException thrown in Java?
12. Why do we sometime get ConcurrentModificationException during
13. iteration?
14. How will you handle InterruptedException in Java?
15. Let say there is a method that throws NullPointerException in the
16. superclass. Can we override it with a method that throws
17. RuntimeException?
18. Can you catch an exception thrown by another thread in Java?
19. When do you use Exception or Error in Java? What is the difference
20. between these two?
21. What happens to the Exception object after the exception handling is
22. done?
23. How will you handle runtime exceptions in JSP?
24. How will you handle exceptions in Spring MVC Framework?
25. What is an exception in Java?
26. What is the difference between checked and unchecked exceptions in Java?
27. What is the purpose of the `try-catch` block?
28. Explain the `finally` block in Java.
29. How is an exception propagated through the call stack in Java?
30. Can you give an example of a custom exception class in Java?
31. What is the difference between `throw` and `throws` in Java?
32. throw` is used to explicitly throw an exception within a method, whereas `throws` is used in a method declaration to specify the exceptions that might be thrown by that method.
33. What is the `try-with-resources` statement in Java, and when is it typically used?
34. How can you prevent a specific piece of code from throwing an exception in Java?
35. What is the purpose of the `try` block without a `catch` block in Java?
36. What is the difference between `System.exit()` and throwing an exception to terminate a Java program?
37. Explain the differences between the `try-catch` and `try-finally` blocks in Java.
38. What is the purpose of the `throws` clause in a method signature?
39. What is the difference between `NullPointerException` and
40. What is the role of the `printStackTrace()` method in exception handling?
41. Explain the concept of chained exceptions in Java.
42. How can you handle multiple exceptions in a single `catch` block?
43. What is the difference between the `throw` and `throws` keywords in Java?
44. How can you create a custom unchecked exception in Java?
45. When should you use checked exceptions, and when should you use
46. What is the purpose of the try-with-resources statement in Java, and how does it improve resource management?
47. Explain the differences between sError and Exception` in Java. When should you catch or handle each of them?
48. How can you rethrow an exception in Java, and what is the purpose of doing so?
49. What is the difference between `Error` and `RuntimeException` in Java, and when should you catch or throw each of them?
50. What is the `try-with-resources` statement in Java, and how does it relate to the `AutoCloseable` interface?
51. How can you create a custom checked exception in Java, and when might it be useful to do so?
52. What is the purpose of the `try-catch-finally` block in Java, and in what scenarios is it commonly used?
53. Can you explain the `catch` or `throws` order of exception handling in multi-level exception hierarchies?
54. What is the difference between a compile-time error and a runtime exception in Java?
55. How do you handle exceptions when working with multi-threading or concurrency in Java?
56. What are the best practices for exception handling in Java, and how can you ensure robust and effective exception handling in your code?
57. Explain the concept of custom exception chaining and its advantages in software development.
58. How do you handle exceptions when working with I/O operations in Java, such as reading or writing to files?
59. Can you provide an example of the `try-catch-resource` pattern in Java, and explain its benefits in resource management?
60. What is the difference between the `finally` block and the `final` keyword in Java?
61. How can you create a custom exception class that includes additional information, such as custom error messages or error codes?
62. Explain the difference between the `throws` and `throw` keywords in Java, and when to use each of them.
63. How do you handle exceptions in Java when dealing with database operations, such as connecting to a database or executing SQL queries?
    1. hat are different scenarios causing “Exception in thread main”?
64. What is OutOfMemoryError in Java?
65. What is the difference between ClassNotFoundException and NoClassDefFoundError?
66. What do you understand by an unreachable catch block error?
67. Is it possible to throw checked exceptions from a static block?
68. What are different scenarios where “Exception in thread main” types of error could occur?