Java Exception Handling – Examples & Explanations

1. Methods to Print Exception Information

```
public class MainClass {
  public static void main(String[] args) {
    int a = 20;
    int b = 0;

    try {
      int c = a / b; // Will throw ArithmeticException
      System.out.println("Answer: " + c);
    }
    catch (Exception e) {
      System.out.println("Using getMessage(): " + e.getMessage());
      System.out.println("Using toString(): " + e.toString());
      System.out.print("Using printStackTrace(): ");
      e.printStackTrace();
    }
    finally {
      System.out.println("Final block executed");
    }
  }
}
```

2. Inner Try-Catch-Finally

```
public class InnerTryCatchExample {
  public static void main(String[] args) {
    int a = 20;
    int b = 0;
```

```
        try {
            // Inner try-catch-finally block
            try {
                int c = a / b; // ArithmeticException
                System.out.println("Answer: " + c);
            }
            catch (ArithmeticException e) {
                System.out.println("Inner Catch: " + e.getMessage());
            }
            finally {
                System.out.println("Final inner block executed");
            }

            // This will cause NullPointerException
            String name = null;
            System.out.println(name.length());
        }
        catch (NullPointerException e) {
            System.out.println("Outer Catch: " + e.toString());
        }
        finally {
            System.out.println("Final outer block executed");
        }
    }
}
```

3. Multiple Catch Blocks

```
public class MultipleCatchExample {
```

```
  public static void main(String[] args) {
    try {
      String name = null; // Will cause NullPointerException
      System.out.println(name.length());
    }
    catch (ArithmeticException e) {
      System.out.println("Arithmetic Exception: " + e.toString());
    }
    catch (ArrayIndexOutOfBoundsException e) {
      System.out.println("Array Index Out of Bounds: " + e.toString());
    }
    catch (NullPointerException e) {
      System.out.println("Null Pointer Exception: " + e.toString());
    }
    finally {
      System.out.println("Final block executed");
    }
  }
}
```

4. throw Keyword

Purpose: Used to explicitly throw an exception object.

Where Used: Inside a method or block.

Effect: Immediately transfers control to the nearest matching catch.

```
public class ThrowExample {
  public static void main(String[] args) {
```

```
    int age = 15;
    if (age < 18) {
      throw new IllegalArgumentException("Age must be 18 or older to vote.");
    }
    System.out.println("Eligible to vote.");
  }
}
```

5. throws Keyword

Purpose: Declares exceptions a method might throw.

Where Used: In method declaration.

Effect: Caller must handle or declare it.

```
import java.io.*;

public class ThrowsExample {
  public static void main(String[] args) {
    try {
      readFile();
    }
    catch (IOException e) {
      System.out.println("Caught: " + e);
    }
  }

  static void readFile() throws IOException {
    FileReader file = new FileReader("test.txt"); // File might not exist
```

```
    BufferedReader fileInput = new BufferedReader(file);
    throw new IOException("Custom IO Error occurred");
  }
}
```

6. Combining throw and throws

```
public class Bank {
  public static void main(String[] args) {
    try {
      withdraw(500);
    }
    catch (Exception e) {
      System.out.println("Error: " + e.getMessage());
    }
  }

  static void withdraw(int amount) throws Exception {
    int balance = 300;
    if (amount > balance) {
      // throw keyword
      throw new Exception("Insufficient balance!");
    }
    System.out.println("Withdraw successful!");
  }
}
```