```
class CountryNotValidException extends Exception {
   public CountryNotValidException(String message) {
      super(message);
   }
}
class EmployeeNameInvalidException extends Exception {
   public EmployeeNameInvalidException(String message) {
      super(message);
   }
}
class TaxNotEligibleException extends Exception {
   public TaxNotEligibleException(String message) {
      super(message);
   }
}
```

```
class TaxCalculator {
  public double calculateTax(String empName, boolean isIndian, double empSal)
      throws CountryNotValidException, EmployeeNameInvalidException, TaxNotEligibleException {
      if (empName == null || empName.isEmpty()) {
          throw new EmployeeNameInvalidException("The employee name cannot be empty");
      }
      if (!isIndian) {
          throw new CountryNotValidException("The employee should be an Indian citizen for calculating tax.");
      }
      if (empSal > 100000) {
          return empSal * 8 / 100;
      } else if (empSal >= 50000 && empSal <= 100000) {
          return empSal * 6 / 100;
      } else if (empSal >= 30000 && empSal < 50000) {
          return empSal * 5 / 100;
      } else if (empSal >= 10000 && empSal < 30000) {
          return empSal * 4 / 100;
      } else {
          throw new TaxNotEligibleException("The employee does not need to pay tax");
      }
  }
}
```

```
public class CalculatorSimulator {
  public static void main(String[] args) {
    TaxCalculator taxCalculator = new TaxCalculator();
    // Test Case 1: Employee Ron, salary 34000, not Indian
    try {
      System.out.println("Test Case 1:");
      double tax = taxCalculator.calculateTax("Ron", false, 34000);
      System.out.println("Tax amount is " + tax);
    } catch (Exception e) {
      e.printStackTrace();
      System.out.println(e.getMessage());
    }
    System.out.println();
    // Test Case 2: Employee Tim, salary 1000, Indian
    try {
      System.out.println("Test Case 2:");
      double tax = taxCalculator.calculateTax("Tim", true, 1000);
      System.out.println("Tax amount is " + tax);
    } catch (Exception e) {
      e.printStackTrace();
      System.out.println(e.getMessage());
    }
    System.out.println();
```

```
    // Test Case 3: Employee Jack, salary 55000, Indian
    try {
        System.out.println("Test Case 3:");
        double tax = taxCalculator.calculateTax("Jack", true, 55000);
        System.out.println("Tax amount is " + tax);
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
    System.out.println();

    // Test Case 4: No name, salary 30000, Indian
    try {
        System.out.println("Test Case 4:");
        double tax = taxCalculator.calculateTax("", true, 30000);
        System.out.println("Tax amount is " + tax);
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println(e.getMessage());
    }
  }
}
```

Question 1 Output -

Test Case 1:
CountryNotValidException: The employee should be an Indian citizen for calculating tax.
The employee should be an Indian citizen for calculating tax.

Test Case 2:
TaxNotEligibleException: The employee does not need to pay tax
The employee does not need to pay tax

Test Case 3:
Tax amount is 3300.0

Test Case 4:
EmployeeNameInvalidException: The employee name cannot be empty
The employee name cannot be empty

```
public class ArrayIndexOutOfBoundsDemo {
  public static void main(String[] args) {
    int[] numbers = {10, 20, 30, 40, 50};
    try {
      for (int i = 0; i <= numbers.length; i++) {
        System.out.println("Element at index " + i + ": " + numbers[i]);
      }
    } catch (ArrayIndexOutOfBoundsException e) {
      System.out.println("Exception caught: " + e);
      System.out.println("Attempted to access an index that is out of bounds.");
    }
  }
}
```

Question 2 Output -

Element at index 0: 10

Element at index 1: 20

Element at index 2: 30

Element at index 3: 40

Element at index 4: 50

Exception caught: java.lang.ArrayIndexOutOfBoundsException: Index 5 out of bounds for length 5

Attempted to access an index that is out of bounds.