The task aims to practice error and exception handling in Java while working with statistical calculations and factorial computation using arrays. The program will collect ages, validate inputs, calculate statistics, and handle different types of exceptions.

Technical Requirements:

1. Main Functionalities:

1. **Age Input and Storage:**
   - The user should input a maximum of **50 ages**, which will be stored in an `int[]` array.
   - The entered ages should be validated for the following conditions:
     - If the input is not a valid number (e.g., a letter or symbol), a `NumberFormatException` should be thrown, and an appropriate message should be displayed.
     - If the entered age is negative, an `IllegalArgumentException` should be thrown with the message "Age cannot be negative."
     - If the entered age is greater than 200, an `ArithmeticException` should be thrown with the message "Age is too large."
2. **Exit Condition:**
   - The user should be able to stop entering ages by typing "stop" or "exit".

---

2. Additional Functionalities:

1. **Factorial Calculation:**
   - For each age entered, the program should calculate the factorial.
   - If the factorial is too large to fit within the range of `long` or `int` data types, an appropriate exception should be thrown and handled.
2. **Check for Even/Odd:**
   - For each age entered, the program should check if the age is even or odd and display the result.

---

3. Statistical Calculations:

The program should calculate and display the following statistics at the end:

1. **Total Number of People:**
   - The total number of ages entered.
2. **Sum of Ages:**
   - The total sum of all the entered ages.
3. **Average of Ages:**

    - **Formula:**Average=Sum of AgesTotal Number of PeopleAverage=Total Number of PeopleSum of Ages
4. **Age Group Statistics:**
    - **0-12:** Child
    - **13-19:** Teenager
    - **20-64:** Adult
    - **65 and above:** Senior
    - The program should display the count of people in each age group.

---

4. Example Outputs:

Inputs:

mathematica
Copy code

```
Enter an age: 5
Enter an age: 20
Enter an age: 65
Enter an age: stop
```

Output:

yaml
Copy code

```
Entered Ages: [5, 20, 65]

Details for each age:
- Age: 5 -> Factorial: 120, Odd.
- Age: 20 -> Factorial: 2432902008176640000, Even.
- Age: 65 -> Factorial too large to calculate, Odd.

Statistical Summary:
- Total number of people: 3
- Sum of ages: 90
- Average age: 30.0
- Age Group Statistics:
  * Children: 1 person
  * Adults: 1 person
  * Seniors: 1 person
```

Suggested Method Structure:

1. `calculateFactorial(int age)`: Method to calculate the factorial.
    - If the factorial exceeds the range of `long`, an exception is thrown.
2. `isEven(int age)`: Method to check if the age is even or odd.
3. `getStatistics(int[] ages, int count)`: Method to calculate the statistical summary (sum, average, and age group counts).