Assignment: Java Lambda and Functional Programming

Objective: To understand and apply Java's lambda expressions and functional programming paradigms in real-world scenarios.

Instructions:

1. Ensure you have the latest version of Java and a suitable IDE (like IntelliJ IDEA or Eclipse).
2. Familiarize yourself with Java's functional interfaces in the **java.util.function** package.

Part 1: Lambda Expressions

1. **Basic Lambda Expressions**:
    - Write a lambda expression that takes two integers and returns their sum.
    - Write a lambda expression that takes a string and returns its uppercase version.
2. **Comparator Using Lambda**:
    - Given a list of strings, sort them based on their length using a lambda expression.
3. **Runnable Using Lambda**:
    - Create a new thread using the Runnable interface and lambda expressions. The thread should print "Lambda Runnable in action!" when run.

Part 2: Functional Programming

1. **Streams**:
    - Convert a list of integers into a stream, filter out the odd numbers, and collect the result into a new list.
2. **Map-Filter-Reduce**:
    - Using a list of strings, convert all strings to uppercase (map), filter out strings that are less than 4 characters long, and concatenate the remaining strings (reduce).
3. **Function Composition**:
    - Create two **Function<Integer, Integer>** definitions: a. One that multiplies the given number by 2. b. Another that adds 3 to the given number.
    - Compose the two functions into a new function that multiplies a given number by 2 and then adds 3.

Part 3: Using Pre-defined Functional Interfaces

1. **Predicates**:
    - Write a predicate that checks if a number is even.
    - Write a predicate that checks if a string's length is greater than 5.
    - Combine the two predicates to check a list of strings and filter out those that are even in length and have a length greater than 5.
2. **Function Interface**:
    - Write a function that takes a string and returns its length.
    - Write a function that takes a string and returns its lowercase version.
3. **Consumer and Supplier**:
    - Write a consumer that prints the string it receives.

- Write a supplier that returns the current date-time as a formatted string.

Submission:

1. Submit a zip file containing all your Java source code files.
2. Include a README.md file explaining your approach for each task.
3. Make sure your code is well-commented and follows Java coding standards.

Evaluation Criteria:

1. Code correctness and efficiency.
2. Adherence to functional programming paradigms.
3. Code readability and organization.
4. Understanding of Java's lambda and functional interfaces.

Bonus Challenge: Implement a mini project where you design a simple address book. Use lambda and functional programming concepts to add, remove, search, and list contacts. Each contact can have a name and phone number. Implement features to search by name, list all contacts, etc.
