Java Consumer Functional Interface - Assignment

Instructions:

You are provided with the `LambdaConsumerExample.java` class. Your task is to implement the `main()` method using Consumer functional interfaces and lambda expressions. Below are the specific tasks you need to complete, including exact variable names and type declarations.

Task 1: Create a Consumer to Print Each Element

Objective: Use a lambda expression to implement a Consumer that prints each element in a list.

Details:
- The task is to print each name from a list prefixed with the word "Element: ".
- You should use the `Consumer<String>` interface for this task.

Steps:
- Create a variable named `printElement` of type `Consumer<String>`.
- Use a lambda expression to implement the `accept()` method.
- Inside the lambda expression, print the message in the format:

```
Element: <name>
```

Task 2: Create and Populate an ArrayList

Objective: Create an `ArrayList<String>` and add elements to it.

Details:
- Create an `ArrayList<String>` that contains the following names:

```
"Alice", "Bob", "Charlie", "Dave"
```

Steps:
- Create and initialize an `ArrayList<String>` with the names listed above.

Task 3: Use the Consumer to Print Elements in the List

Objective: Use the previously created `printElement` Consumer to print each element in the `ArrayList`.

Details:
- Use the `forEach()` method to iterate through the list and apply the `printElement` Consumer.

Task 4: Create Another Consumer to Print Elements in Uppercase

Objective: Create another `Consumer<String>` to transform and print elements in uppercase.

Details:
- Create a new `Consumer<String>` variable named `printUppercase`.
- Use a lambda expression to print each name in uppercase format with the word "Uppercase: " prefixed.

Steps:
- Create a variable named `printUppercase` of type `Consumer<String>`.
- Use a lambda expression to transform and print each name to uppercase.

Task 5: Use the Second Consumer to Print Elements in Uppercase

Objective: Use the `printUppercase` Consumer to print all elements in the `ArrayList`.

Details:
- Again use the `forEach()` method to apply the `printUppercase` Consumer.

Final Deliverable:

Implement the `main()` method in the `LambdaConsumerExample.java` class. Ensure that:
- You declare and initialize the `ArrayList<String>` with the names "Alice", "Bob", "Charlie", and "Dave".
- You create a `Consumer<String>` variable named `printElement` to print elements with the "Element:" prefix.
- You create a `Consumer<String>` variable named `printUppercase` to print elements in uppercase with the "Uppercase:" prefix.
- You use the `forEach()` method to apply both Consumers.

Do not include any print statements outside of the lambda expressions.

Execution Steps to Follow:

1. **All actions like build, compile, running application, running test cases will be through Command Terminal.**

2. **To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) □ Terminal □New Terminal.**

3. **This editor Auto Saves the code.**

4. **If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.**

5. **These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.**

6. **To run your project use command:**

   ```
   mvn compile exec:java
   -Dexec.mainClass="com.yaksha.assignment.LambdaConsumerExample"
   ```

7. **To test your project test cases, use the**

   ```
   mvn test
   ```

8. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**