Java Lambda Expressions - Assignment

Instructions:

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

Task 1: Create a Lambda Expression to Print a Message

1. **Objective**: Use a lambda expression to implement a simple task that prints a message.

2. **Details**:
   - The task is to print the following message to the console:
     ```
     "Hello from Lambda Expression!"
     ```
   - You should use the `Runnable` interface for this task.

3. **Steps**:
   - **Create a variable** named `printMessage` of type `Runnable`.
   - **Use a lambda expression** to implement the `run()` method of the `Runnable` interface.
   - Inside the lambda expression, **print the message** `"**Hello from Lambda Expression!**"`.

Task 2: Use Lambda Expression with an ArrayList

1. **Objective**: Use a lambda expression to create an `ArrayList` of names.

2. **Details**:
   - Create an `ArrayList<String>` that contains the following names:
     ```
     "John", "Jane", "Mark", "Paul"
     ```

- You need to perform one action with the list of names:
  - **Action 1**: **Create and initialize** an `ArrayList<String>` with the names `"John"`, `"Jane"`, `"Mark"`, and `"Paul"`.

Task 3: Use Predicate for Filtering Names

1. **Objective**: Use the `Predicate<String>` functional interface to filter the names.

2. **Details**:
   - You have to use Predicate to filter names based on a condition.
   - You need to create a `Predicate<String>` that checks if a name starts with the letter `"J"`.

3. **Steps**:
   - **Create a `Predicate<String>` variable** named `startsWithJ`.
   - **Use a lambda expression** to define the condition: the name should start with `"J"`.
   - Use the `filter()` method of the `stream()` to **apply the predicate** and filter the names in the `ArrayList` that satisfy the condition.
   - After filtering, **print each name** that starts with `"J"`.

Final Deliverable:

1. Implement the `main()` method in the `LambdaExpressionExample.java` class.
2. Ensure that:
   - You **declare and initialize** the `ArrayList<String>` with the names `"John"`, `"Jane"`, `"Mark"`, and `"Paul"`.
   - You **create a `Runnable` variable** named `printMessage` to print the message `"Hello from Lambda Expression!"`.
   - You **create a `Predicate<String>` variable** named `startsWithJ` to filter names starting with `"J"`.
   - You use **appropriate lambda expressions** for each task.
   - You **print filtered names** (those starting with `"J"`) using the `forEach()` method.
3. Do not include any print statements outside of the lambda expressions.

Once you have completed your implementation, save your changes and submit the Java file.

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.LambdaExpressionExample"

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

   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.**