Programación Avanzada

Taller grupal 3

Este taller tiene como objetivo usar *jmh* para medir el desempeño de los tres algoritmos para encontrar números primos en las listas de números que se crearon en la clase pasada, además, deberán realizar una comparación con los datos que obtuvieron anteriormente. No olvide que su nuevo programa debe tomar en cuenta la fórmula revisada en clases para determinar el número de hilos.

Le recomiendo crear un nuevo proyecto y copiar el código que desarrolló anteriormente, una vez hecho esto, implemente lo necesario para usar *jmh* para medir el desempeño, usted debe seleccionar el modo del *benchmark* que le permita comparar los nuevos datos con los anteriores.

Si usted revisa el repositorio del *jmh* (https://github.com/openjdk/jmh) usted encontrar algunas recomendaciones para trabajar de la mejor manera con esta librería, una de ellas es el contenido del archivo *pom.xml*, se recomienda usar lo siguiente (aquí únicamente se presenta una parte):

```
<properties>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  <maven.compiler.source>22</maven.compiler.source>
  <maven.compiler.target>22</maven.compiler.target>
  <uberjar.name>benchmarks</uberjar.name>
</properties>

<dependencies>
  <dependency>
    <groupId>org.openjdk.jmh</groupId>
    <artifactId>jmh-core</artifactId>
    <version>1.37</version>
  </dependency>
  <dependency>
    <groupId>org.openjdk.jmh</groupId>
    <artifactId>jmh-generator-annprocess</artifactId>
    <version>1.37</version>
  </dependency>
</dependencies>
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.13.0</version>
      <configuration>
      </configuration>
    </plugin>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-surefire-plugin</artifactId>
      <version>3.2.5</version>
      <configuration>
        <redirectTestOutputToFile>true</redirectTestOutputToFile>
      </configuration>
    </plugin>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-shade-plugin</artifactId>
      <version>3.5.3</version>
      <executions>
        <execution>
          <phase>package</phase>
          <goals>
            <goal>shade</goal>
          </goals>
          <configuration>
            <finalName>${uberjar.name}</finalName>
            <transformers>
              <transformer implementation="org.apache.maven.plugins.shade.resource.ManifestResourceTransformer">
                <mainClass>org.openjdk.jmh.Main</mainClass>
              </transformer>
              <transformer implementation="org.apache.maven.plugins.shade.resource.ServicesResourceTransformer"/>
            </transformers>
```

```
          </configuration>
        </execution>
      </executions>
    </plugin>
  </plugins>
</build>
```

Además, la misma documentación recomienda como ejecutar el proyecto y presenta las siguientes indicaciones:

```
/*
* ============================= HOW TO RUN THIS TEST: ====================================
*
*
* a) Via command-line:
*    $ mvn clean install
*    $ java -jar target/benchmarks.jar JMHSample_01
*
* JMH generates self-contained JARs, bundling JMH together with it.
* The runtime options for the JMH are available with "-h":
*    $ java -jar target/benchmarks.jar -h
*
*/
```

Y el método *main* debe tener la siguiente forma:

```
public static void main( String[] args ) RunnerException {
    //org.openjdk.jmh.Main.main(args);
    Options opt = new OptionsBuilder()
            .include(AppSimple.class.getSimpleName())
            .forks(1)
            .build();
    new Runner(opt).run();
}
```

Verifique que entre sus *imports* estén los siguientes:

```
import org.openjdk.jmh.runner.Runner;
import org.openjdk.jmh.runner.RunnerException;
import org.openjdk.jmh.runner.options.Options;
import org.openjdk.jmh.runner.options.OptionsBuilder;
```

Esta forma de ejecución demanda que se tenga instalado y correctamente configurada la última versión de *Maven* (https://maven.apache.org/download.cgi). Usted verificar la instalación en una ventana de comandos ejecutando el comando `mvn ––version`.

Finalmente, para poder ejecutar el programa, con la lista que se carga del archivo deben modificar la clase DataGenerator y usar el siguiente método:

```
public static List<Integer> loadFromFile() throws IOException {
    try(InputStream resource = DataGenerator.class.getClassLoader().getResourceAsStream("5milNums.txt");
        InputStreamReader ioStreamReader = new InputStreamReader(resource, StandardCharsets.UTF_8);
        BufferedReader bufferedReader = new BufferedReader(ioStreamReader)
    ) {
        return bufferedReader.lines()
                .map(Integer::parseInt)
                .toList();
    }
}
```