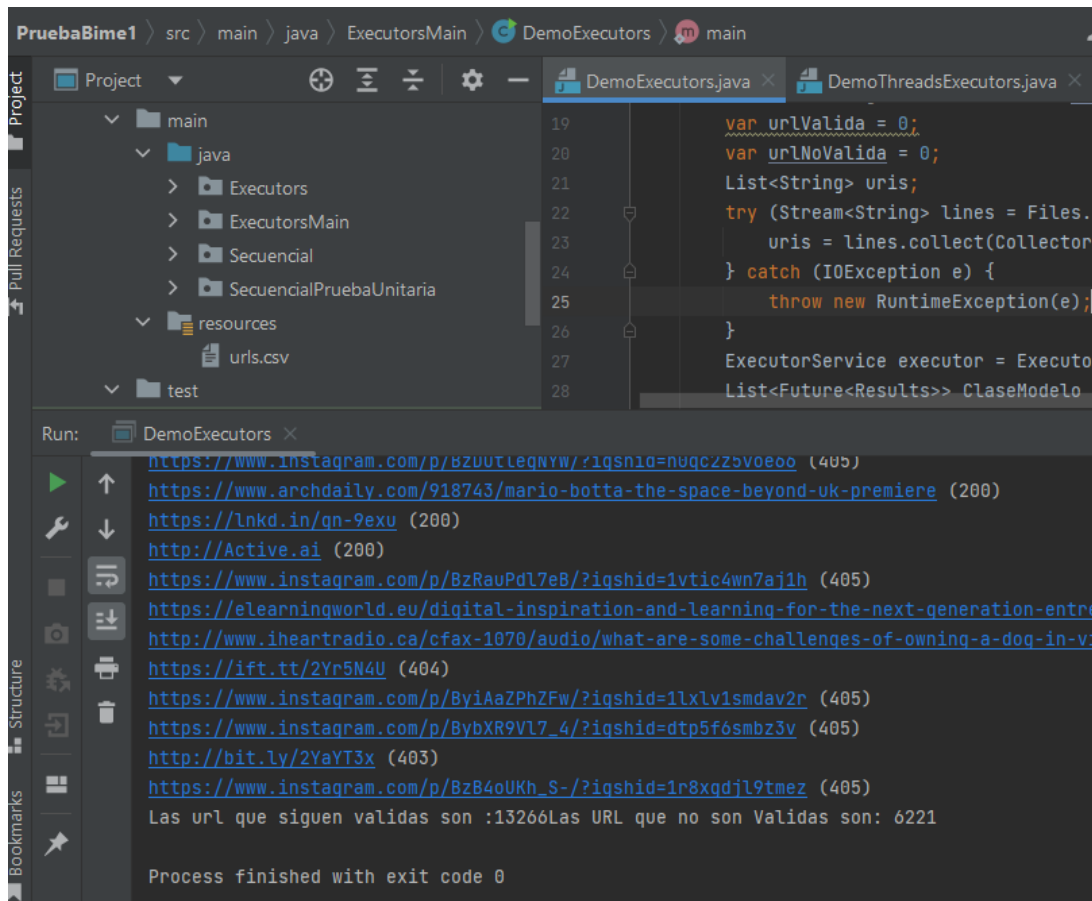


1. Enlace repositorio

<https://github.com/lpuchaicela/PruebaBimestral>

2. ¿Cuántas URLs son válidas (tienen código de estado igual a 200)?



```
19  var urlValida = 0;
20  var urlNoValida = 0;
21  List<String> uris;
22  try (Stream<String> lines = Files.
23      uris = lines.collect(Collectors.
24  } catch (IOException e) {
25      throw new RuntimeException(e);
26  }
27  ExecutorService executor = Executors.
28  List<Future<Results>> ClaseModelo
```

Run: DemoExecutors

```
https://www.instagram.com/p/BZuUtleqNYW/?igsnid=nuqcZz5v0e0o (405)
https://www.archdaily.com/918743/mario-botta-the-space-beyond-uk-premiere (200)
https://lnkd.in/qn-9exu (200)
http://Active.ai (200)
https://www.instagram.com/p/BzRauPdL7eB/?igshid=1vtic4wn7aj1h (405)
https://elearningworld.eu/digital-inspiration-and-learning-for-the-next-generation-entre
http://www.iheartradio.ca/cfax-1070/audio/what-are-some-challenges-of-owning-a-dog-in-vi
https://ift.tt/2Yr5N4U (404)
https://www.instagram.com/p/ByiAaZPhZFw/?igshid=1lxlv1smdav2r (405)
https://www.instagram.com/p/BybXR9VL7_4/?igshid=ntp5f6smbz3v (405)
http://bit.ly/2YaYT3x (403)
https://www.instagram.com/p/BzB4oUKh_S-/?igshid=1r8xqdl9tmez (405)
Las url que siguen validas son :13266Las URL que no son Validas son: 6221
Process finished with exit code 0
```

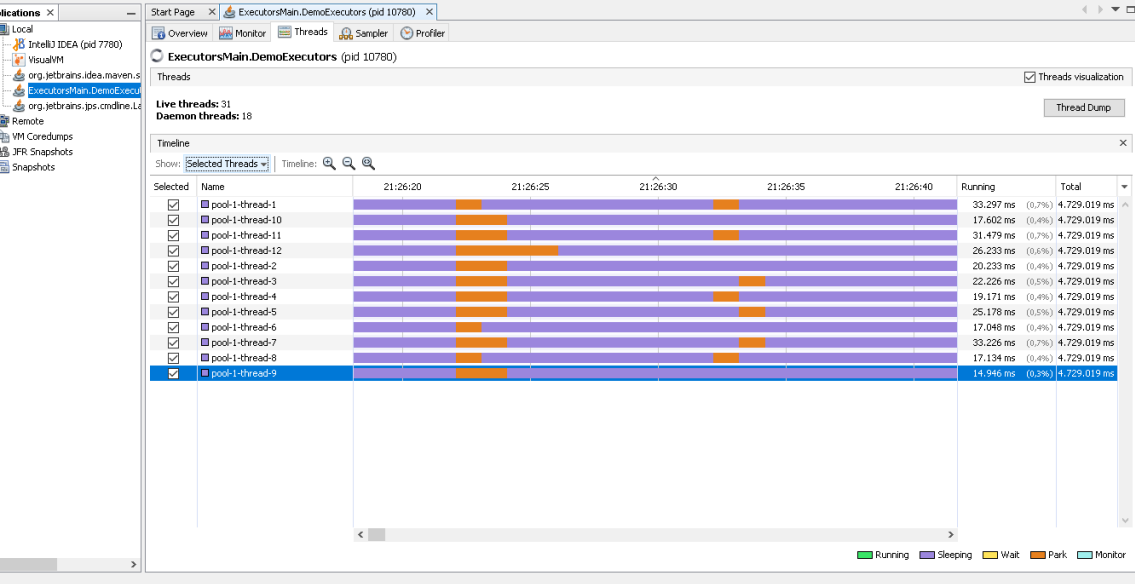
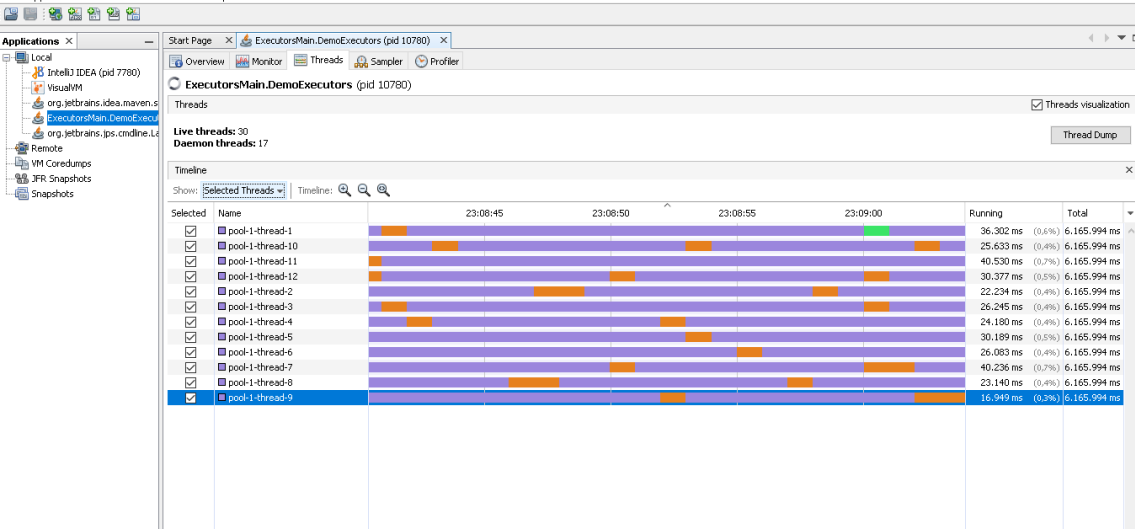
Las URL que son validas con estado igual a 200 son 13.266 urls.

3. ¿Cuántas URLs están rotas o necesitan de una acción adicional (tienen código de estado diferente de 200 o lanzan excepción)?

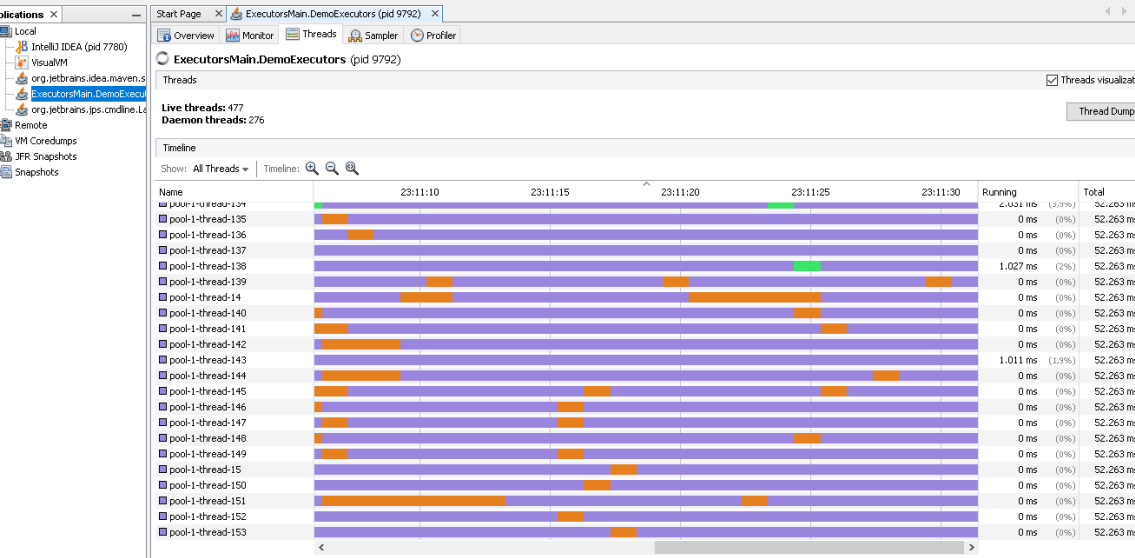
Las URL rotas o invalidas con estado distinto a 200 son 6.221 urls.

4. La imagen de VisualVM o herramienta similar que muestre la ejecución de sus hilos.

Con 14 Hilos

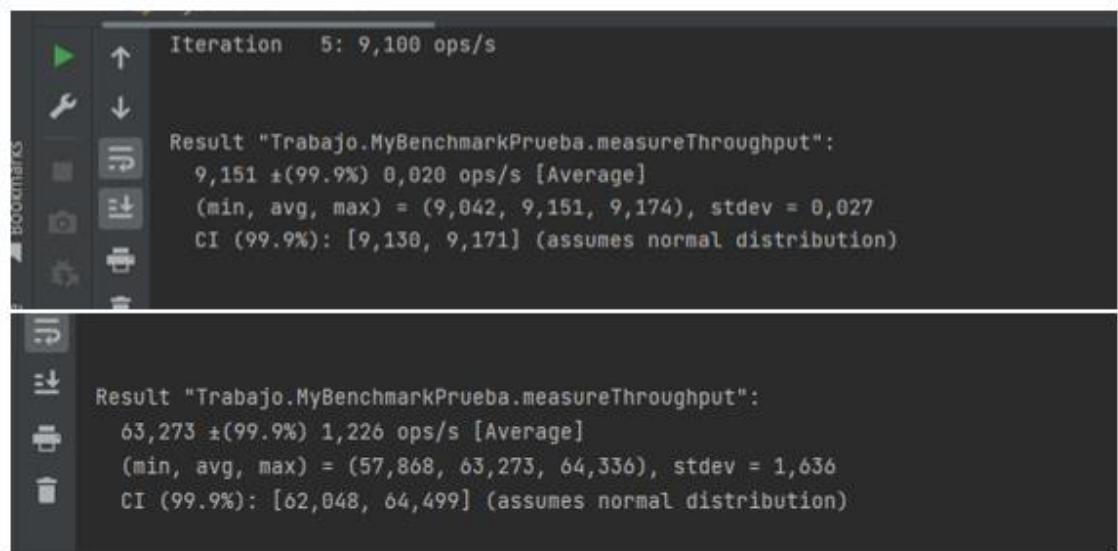


Con 200 Hilos



5. Suba una imagen con el resumen de la ejecución de JMH, la imagen debe ser similar a:

Hilos



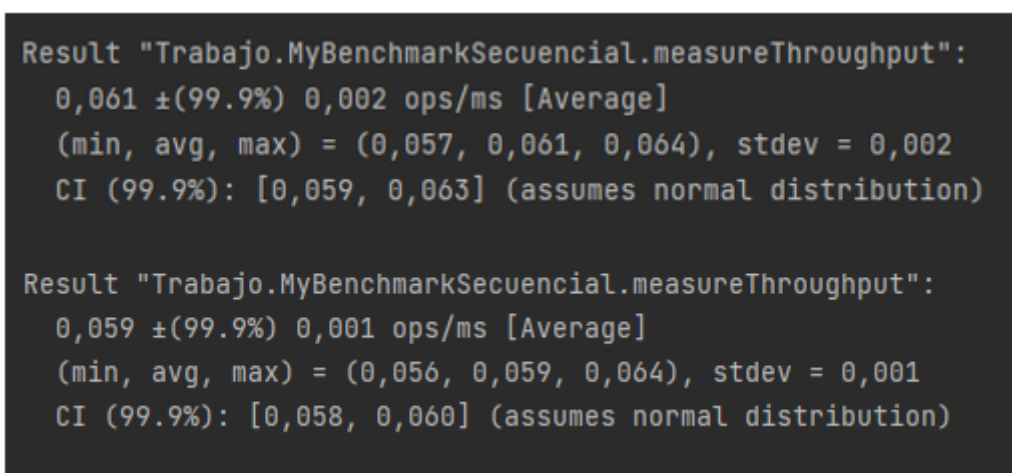
The screenshot shows two panels of JMH results. The top panel displays 'Iteration 5: 9,100 ops/s' and the bottom panel shows the final result for 'Trabajo.MyBenchmarkPrueba.measureThroughput'.

```
Iteration 5: 9,100 ops/s

Result "Trabajo.MyBenchmarkPrueba.measureThroughput":
  9,151 ±(99.9%) 0,020 ops/s [Average]
  (min, avg, max) = (9,042, 9,151, 9,174), stdev = 0,027
  CI (99.9%): [9,130, 9,171] (assumes normal distribution)

Result "Trabajo.MyBenchmarkPrueba.measureThroughput":
  63,273 ±(99.9%) 1,226 ops/s [Average]
  (min, avg, max) = (57,868, 63,273, 64,336), stdev = 1,636
  CI (99.9%): [62,048, 64,499] (assumes normal distribution)
```

Secuencial



The screenshot shows two panels of JMH results for the 'Secuencial' benchmark. The top panel displays the result for 'Trabajo.MyBenchmarkSecuencial.measureThroughput' and the bottom panel shows the final result for 'Trabajo.MyBenchmarkSecuencial.measureThroughput'.

```
Result "Trabajo.MyBenchmarkSecuencial.measureThroughput":
  0,061 ±(99.9%) 0,002 ops/ms [Average]
  (min, avg, max) = (0,057, 0,061, 0,064), stdev = 0,002
  CI (99.9%): [0,059, 0,063] (assumes normal distribution)

Result "Trabajo.MyBenchmarkSecuencial.measureThroughput":
  0,059 ±(99.9%) 0,001 ops/ms [Average]
  (min, avg, max) = (0,056, 0,059, 0,064), stdev = 0,001
  CI (99.9%): [0,058, 0,060] (assumes normal distribution)
```