**Rapport\_Java Performance\_TP2**

**- LIU Yuanyuan && GUO Xiaoqing**

**TP2 : JMH et les micro-benchmarks**

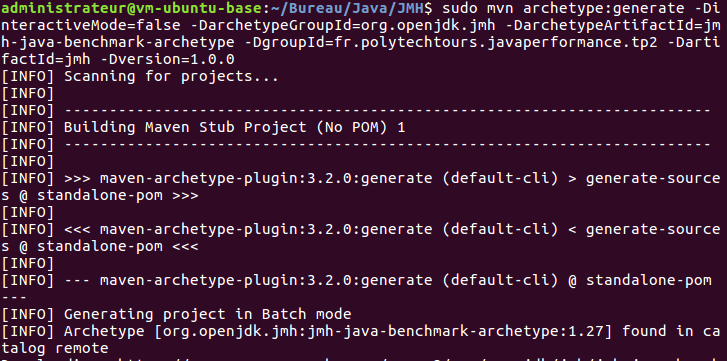
**1.1 Génération et lancement du projet JMH**

La methode de creer un projet JMH est de passer par maven :

command :

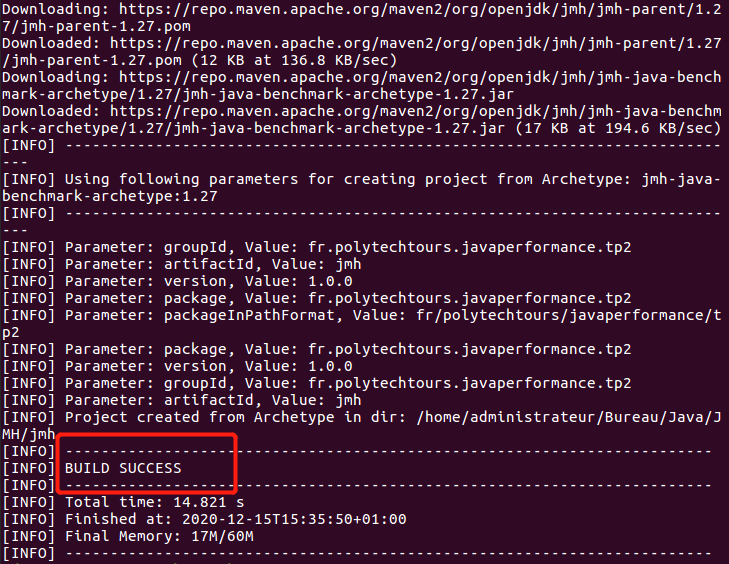
*mvn archetype:generate -DinteractiveMode=false -DarchetypeGroupId=org.openjdk.jmh -DarchetypeArtifactId=jmh-java-benchmark-archetype -DgroupId=fr.polytechtours.javaperformance.tp2 -DartifactId=jmh -Dversion=1.0.0*

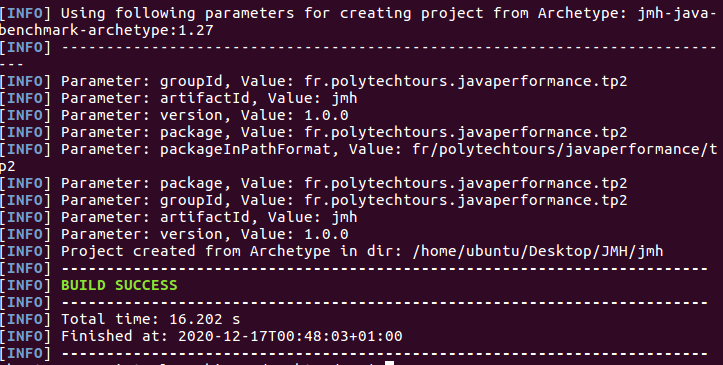
resultat :





... ...







construire le jar :

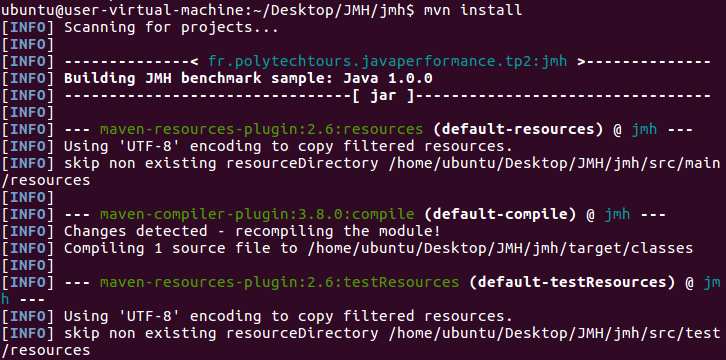
command :

*【1】老师给的命令： mvn clean install 🡺 Build failure*

*【2】更改的命令：mvn install 🡺 Build success*

resultat :

【2】



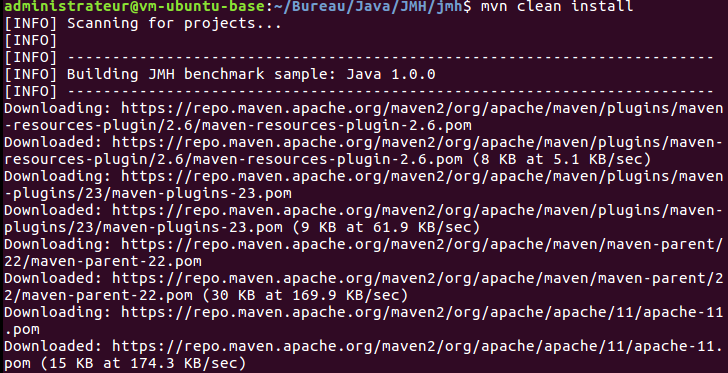
… …



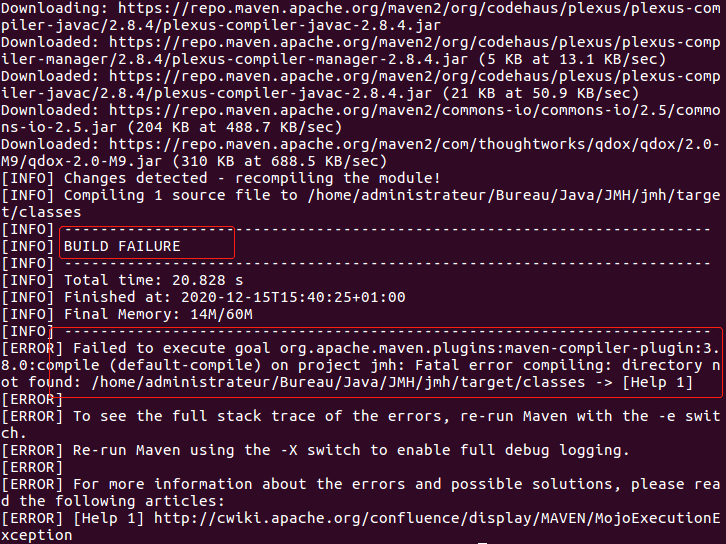




【1】



… …

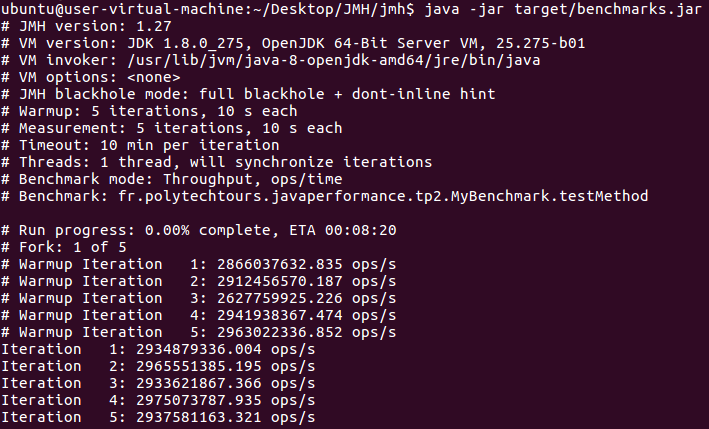


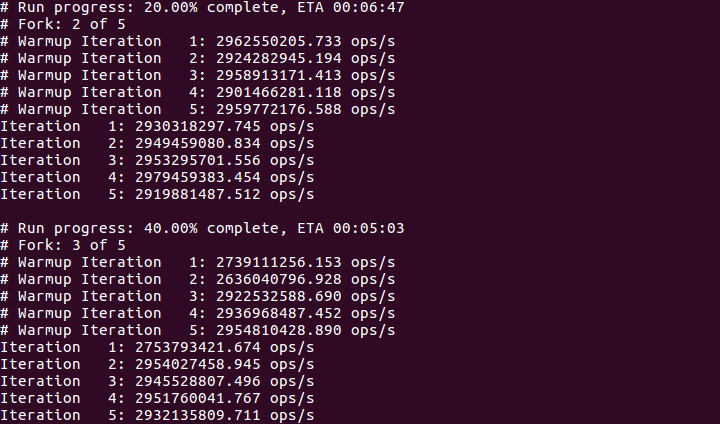


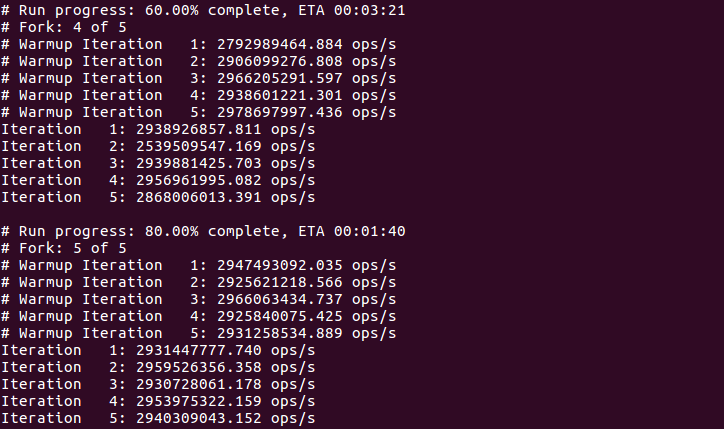
**benchmarks** :

command : *java -jar target/benchmarks.jar*

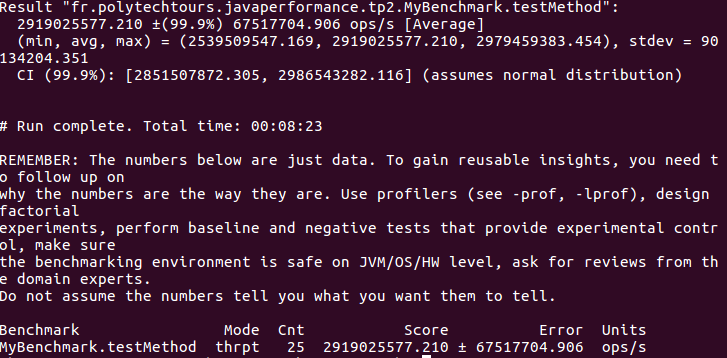
resultat :







A la fin du benchmark, vous obtiendrez un rapport ressemblant au suivant :



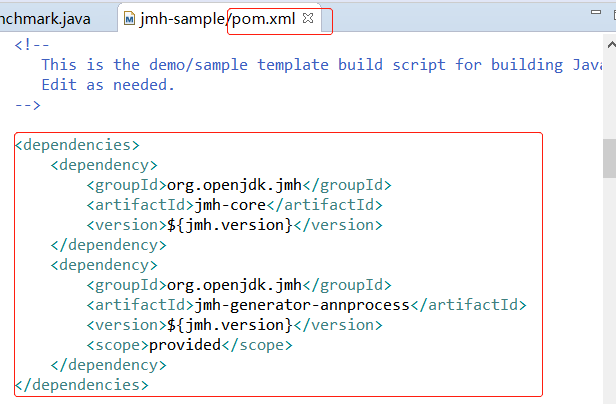
**1.2 Etude de JMH**

简要介绍一下JMH

**1.3 Mise en place d'un micro-benchmark**

项目名称：jmh-sample

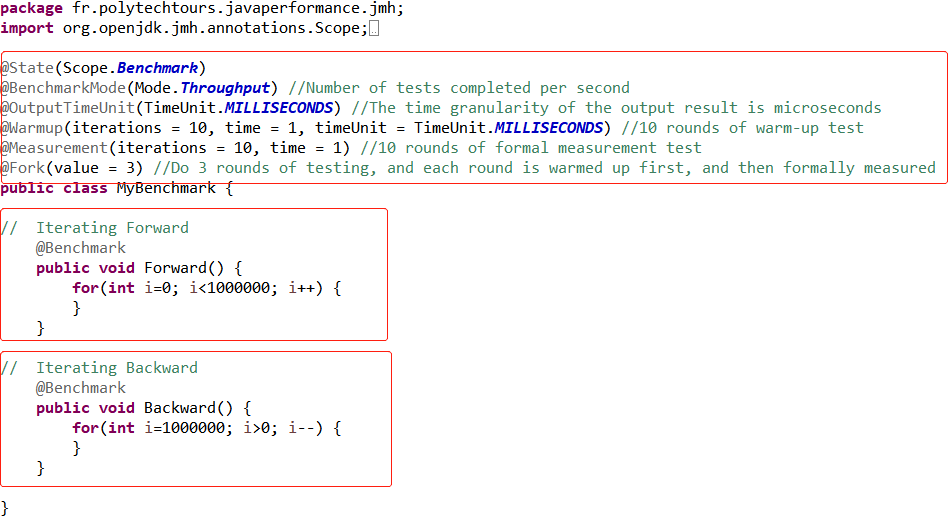
1. 通过pom.xml文件引入jmh依赖



1. 测试 正序/倒序 迭代求和，次数是1百万次迭代

c类名：MyBenchmark

Code :



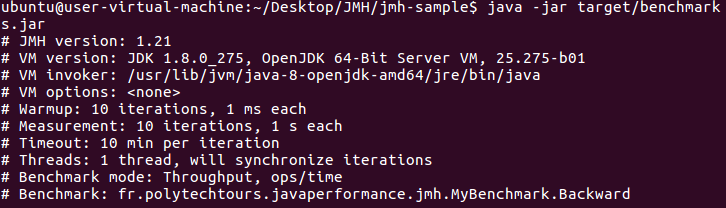
Benchmark配置策略：

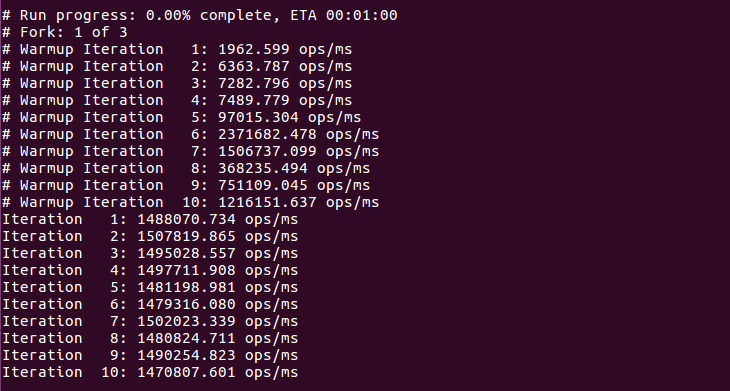
* AverageTim: temps d'exécution moyen de la méthode de test
* Throughput: Nombre de tests effectués par seconde
* TimeUnit.MILLISECONDS: La granularité temporelle du résultat de sortie est de microsecondes
* Warmup: 10 tours de test d'échauffement
* Measurement: 10 tours de test de mesure formel
* Fork: Faites 3 séries de tests, et chaque ronde est d'abord réchauffée, puis mesurée formellement

Forward() : Itérer dans un ordre positif puis additionner

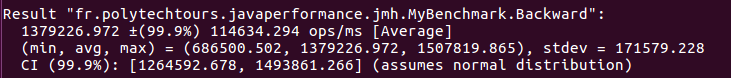
Backward() : Itérer dans l'ordre inverse, puis additionner

1. Benchmark resultat :

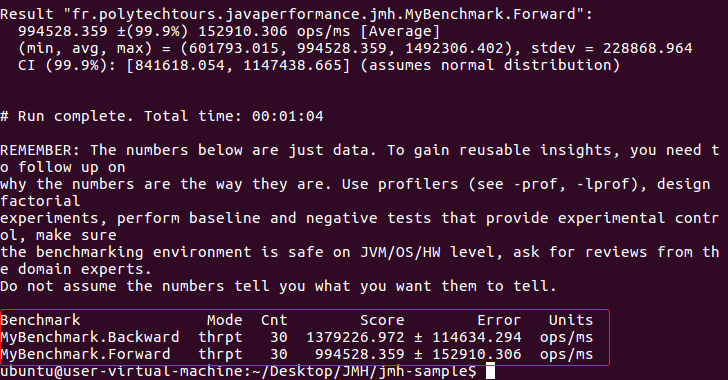




... ...



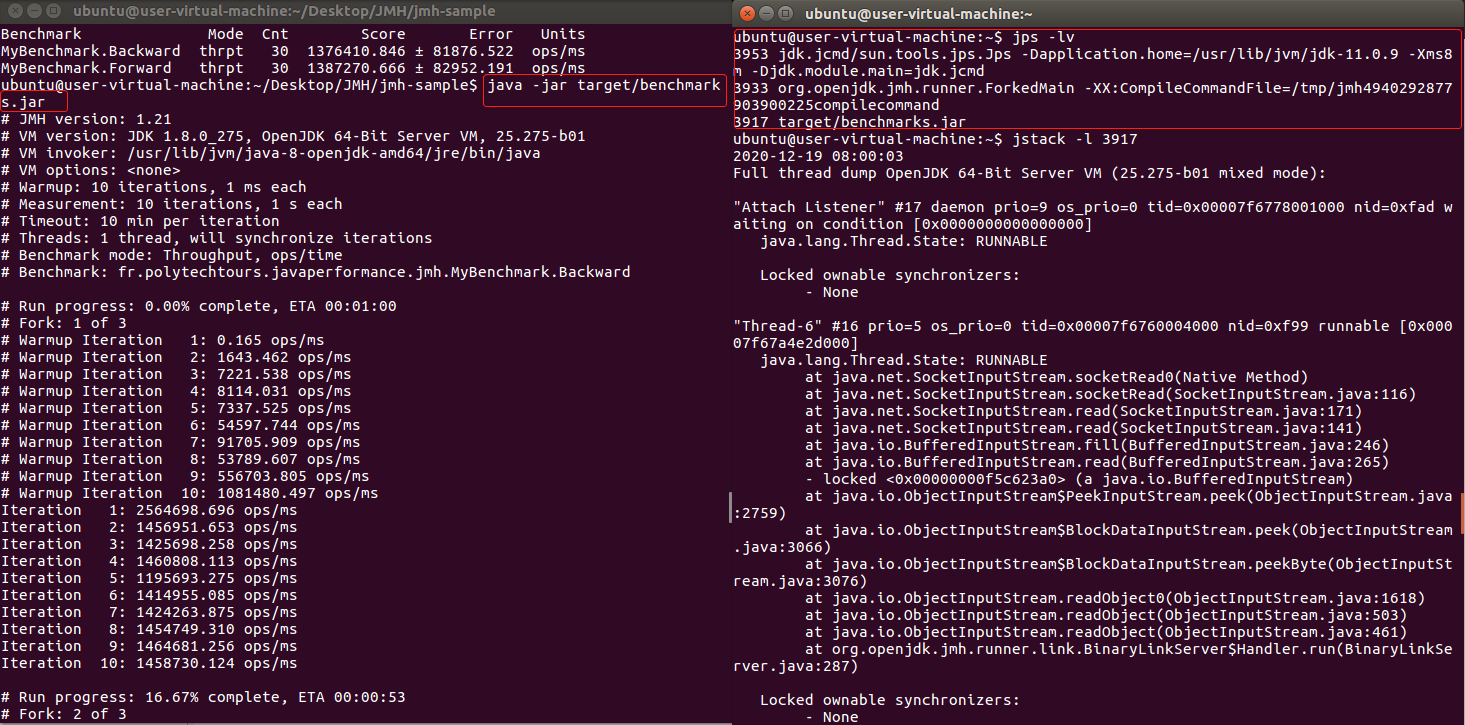
... ...



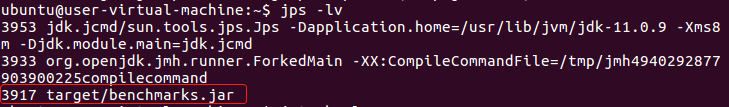
有数据表现可以看到逆序在宏观上是优于正序的。

**1.4 Analyse des methodes testees（选做题，感觉做的不怎么对，最后可以不写）**

我们使用的是JVM ((jvisualvm)来进行分析：(java项目编译运行时同步进行JVM测试)

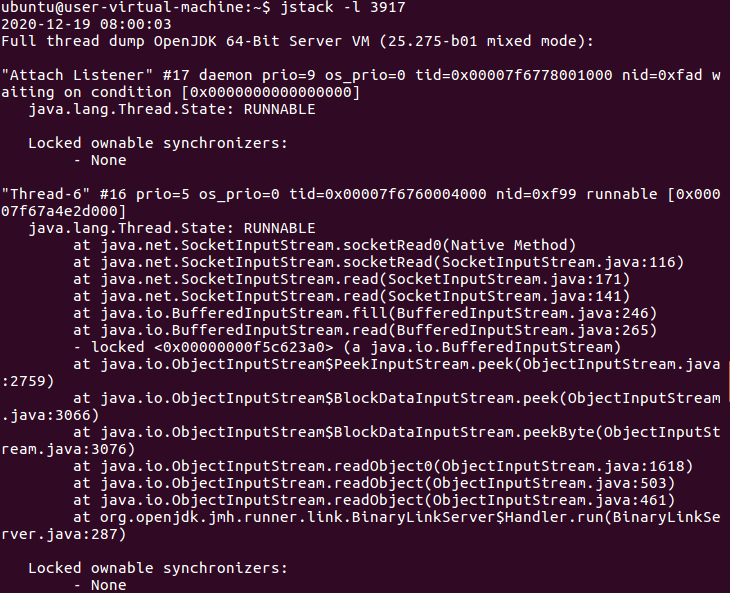


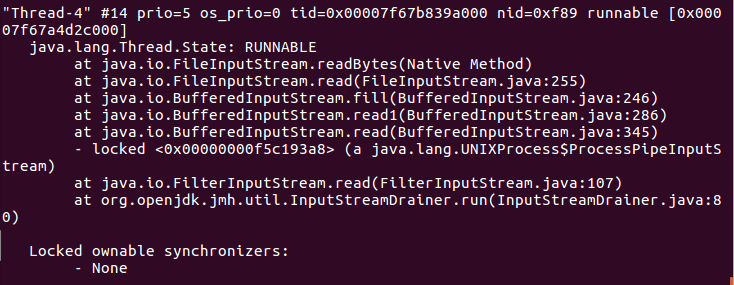
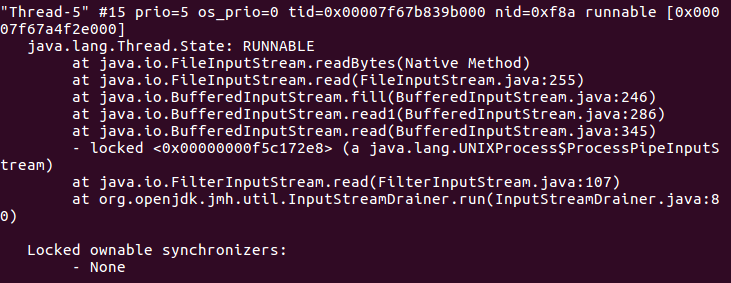
jps –lv : 显示Java进程信息

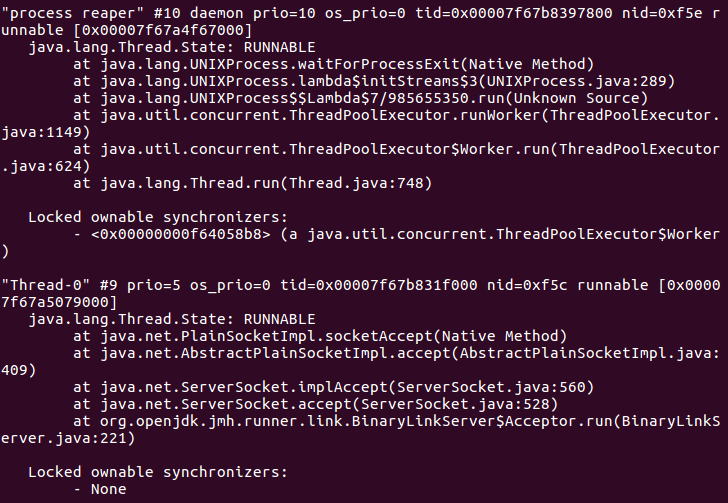


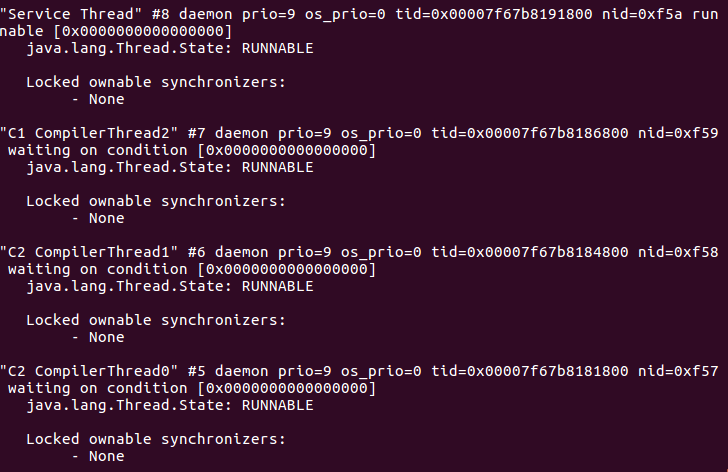
当前进程为3917

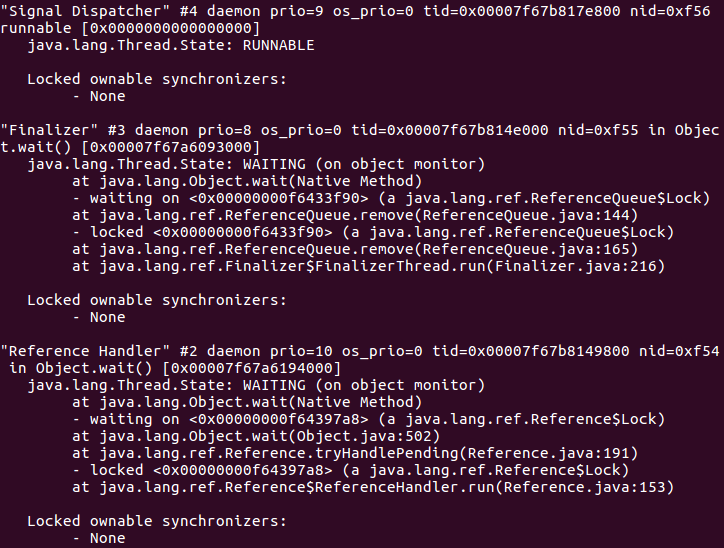
jstack: 查看 Java 进程内当前时刻的线程快照，也就是每条线程正在执行的方法栈情况，用于定位线程停顿、死锁等长时间等待的问题。

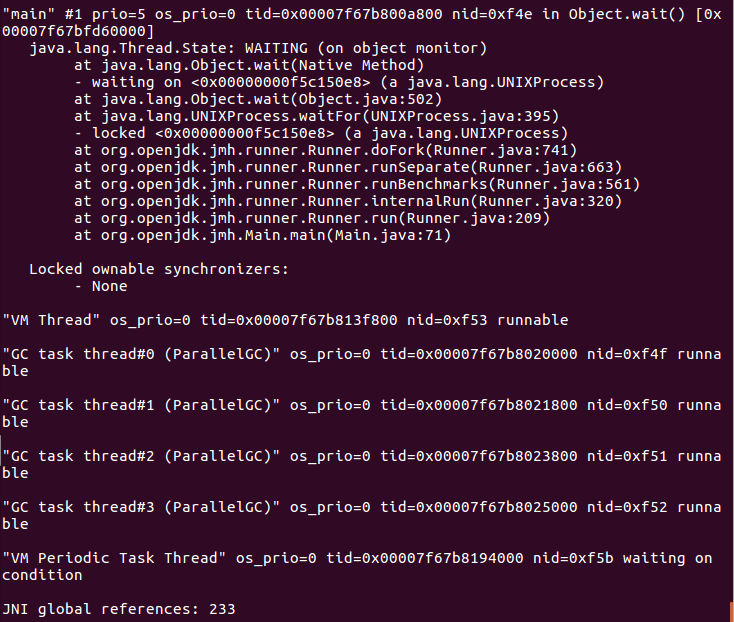












jmap : 查看给定进程、核心文件、远程调试服务器的共享对象内存映射和堆内存细节的工具，可查看堆使用情况、堆内对象直方图、加载类、生成堆快照等

jmap –clstats <pid> : Java 类加载器（ClassLoader）信息，包括加载器名称、已加载类个数、占用空间、父加载器、是否存活、类型信息。

