

Raport laboratorium Akka

Zadanie 1.

```
Z1 x
Started. Commands: 'hi', 'm [nb1] [nb2]', 'd [nb1] [nb2]', 'q'
d 4 2
result: 2(operation count: 1)
m 4 2
result: 8(operation count: 1)
d 4 1
result: 4(operation count: 2)
m 4 1
result: 4(operation count: 2)
d 5 0
[WARN] [05/11/2020 12:05:06.380] [local_system-akka.actor.default-dispatcher-8] [akka://local_system/user/math/divideWorker] / by zero
d 5 4
result: 1(operation count: 4)
m 3 2
result: 6(operation count: 3)
d aaa
[ERROR] [05/11/2020 12:05:46.947] [local_system-akka.actor.default-dispatcher-7] [akka://local_system/user/math/divideWorker] For input string: "aaa"
java.lang.NumberFormatException: For input string: "aaa"
    at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.java:68)
    at java.base/java.lang.Integer.parseInt(Integer.java:658)
    at java.base/java.lang.Integer.parseInt(Integer.java:776)
    at Z1.DivideWorker.Divide(Z1.DivideWorker.java:24)
    at Z1.DivideWorker.lambda$createReceive$0(Z1.DivideWorker.java:14)
    at akka.japi.pf.UnitCaseStatement.apply(CaseStatements.scala:24)
    at akka.japi.pf.UnitCaseStatement.apply(CaseStatements.scala:20)
    at scala.PartialFunction.applyOrElse(PartialFunction.scala:187)
    at scala.PartialFunction.applyOrElse$(PartialFunction.scala:186)
    at akka.japi.pf.UnitCaseStatement.applyOrElse(CaseStatements.scala:20)
    at scala.PartialFunction$OrElse.applyOrElse(PartialFunction.scala:241)
    at akka.actor.Actor.aroundReceive(Actor.scala:535)
    at akka.actor.Actor.aroundReceive$(Actor.scala:533)
    at akka.actor.AbstractActor.aroundReceive(AbstractActor.scala:220)
    at akka.actor.ActorCell.receiveMessage(ActorCell.scala:575)
    at akka.actor.ActorCell.invoke(ActorCell.scala:545)
    at akka.dispatch.Mailbox.processMailbox(Mailbox.scala:270)
    at akka.dispatch.Mailbox.run(Mailbox.scala:231)
    at akka.dispatch.Mailbox.exec(Mailbox.scala:243)
    at java.base/java.util.concurrent.ForkJoinTask.doExec(ForkJoinTask.java:290)
    at java.base/java.util.concurrent.ForkJoinPool$WorkQueue.topLevelExec(ForkJoinPool.java:1016)
    at java.base/java.util.concurrent.ForkJoinPool.scan(ForkJoinPool.java:1665)
    at java.base/java.util.concurrent.ForkJoinPool.runWorker(ForkJoinPool.java:1598)
    at java.base/java.util.concurrent.ForkJoinWorkerThread.run(ForkJoinWorkerThread.java:177)
d 4 3
result: 1(operation count: 1)
m 2 4
result: 8(operation count: 4)
```

Output programu ze strategią OneForOneStrategy

Widzimy, że po wystąpieniu `ArithmeticException` licznik wykonanych operacji nie wyzerował się i policzył operację dzielenia przez 0 również do sumy, a następnie po wznowieniu kontynuował liczenie (czynność `resume()`). Po wystąpieniu innego wyjątku dla operacji `divide` licznik wyzerował się i rozpoczął liczenie od początku (czynność `restart()`). Dla operacji `multiply` licznik nie zmienił się po wystąpieniu drugiego wyjątku (strategia `OneForOne`).

```

Z1 x
Starting JUnit JVM: akka://local_system
Started. Commands: 'hi', 'm [nb1] [nb2]', 'd [nb1] [nb2]', 'q'
d 4 2
result: 2(operation count: 1)
m 4 2
result: 8(operation count: 1)
d 6 3
result: 2(operation count: 2)
m 6 3
result: 18(operation count: 2)
d 4 0
[WARN] [05/11/2020 12:08:10.661] [local_system-akka.actor.default-dispatcher-6] [akka://local_system/user/math/divideWorker] / by zero
d 3 2
result: 1(operation count: 4)
m 3 2
result: 6(operation count: 3)
d aaa
[ERROR] [05/11/2020 12:08:25.665] [local_system-akka.actor.default-dispatcher-6] [akka://local_system/user/math/divideWorker] For input string: "aaa"
java.lang.NumberFormatException: For input string: "aaa"
    at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.java:69)
    at java.base/java.lang.Integer.parseInt(Integer.java:658)
    at java.base/java.lang.Integer.parseInt(Integer.java:776)
    at Z1.DivideWorker.Divide(Z1.DivideWorker.java:24)
    at Z1.DivideWorker.lambda$createReceive$0(Z1.DivideWorker.java:14)
    at akka.japi.pf.UnitCaseStatement.apply(CaseStatements.scala:24)
    at akka.japi.pf.UnitCaseStatement.applyOrElse(CaseStatements.scala:20)
    at scala.PartialFunction.applyOrElse(PartialFunction.scala:187)
    at scala.PartialFunction.applyOrElse$(PartialFunction.scala:186)
    at akka.japi.pf.UnitCaseStatement.applyOrElse(CaseStatements.scala:20)
    at scala.PartialFunction$OrElse.applyOrElse(PartialFunction.scala:241)
    at akka.actor.Actor.aroundReceive(Actor.scala:535)
    at akka.actor.Actor.aroundReceive$(Actor.scala:533)
    at akka.actor.AbstractActor.aroundReceive(AbstractActor.scala:220)
    at akka.actor.ActorCell.receiveMessage(ActorCell.scala:575)
    at akka.actor.ActorCell.invoke(ActorCell.scala:545)
    at akka.dispatch.Mailbox.processMailbox(Mailbox.scala:270)
    at akka.dispatch.Mailbox.run(Mailbox.scala:231)
    at akka.dispatch.Mailbox.exec(Mailbox.scala:243)
    at java.base/java.util.concurrent.ForkJoinTask.doExec(ForkJoinTask.java:290)
    at java.base/java.util.concurrent.ForkJoinPool$WorkQueue.topLevelExec(ForkJoinPool.java:1016)
    at java.base/java.util.concurrent.ForkJoinPool.scan(ForkJoinPool.java:1665)
    at java.base/java.util.concurrent.ForkJoinPool.runWorker(ForkJoinPool.java:1598)
    at java.base/java.util.concurrent.ForkJoinWorkerThread.run(ForkJoinWorkerThread.java:177)

d 3 1
result: 3(operation count: 1)
m 4 3
result: 12(operation count: 1)

```

Program z strategią AllForOneStrategy

Po zmianie strategii zachowanie dla ArithmeticException nie zmieniło się i obaj aktorzy (divide i multiply) kontynuowali liczenie od poprzedniego stanu. Zachowanie zmieniło się po wystąpieniu drugiego wyjątku po którym obaj aktorzy rozpoczęli liczenie od początku (czynność `restart()` i strategia `AllForOne`).

Zadanie 2.

```

Z2_AppLocal x Z2_AppRemote x
[WARN] [05/11/2020 20:55:26.595] [main] [akka.remote.RemoteActorRefProvider] Using the 'remote' ActorRefProvider directly, which is a low-level layer. For most use cases, the 'cluster' abstraction on top of remoting is more suitable instead.
[WARN] [05/11/2020 20:55:26.596] [main] [akka.remote.RemoteActorRefProvider] Akka Cluster not in use - Using Akka Cluster is recommended if you need remote watch and deploy.
[INFO] [05/11/2020 20:55:26.969] [main] [ArteryTcpTransport(akka://local_system)] Remoting started with transport [Artery tcp]; listening on address [akka://local_system@127.0.0.1:2551] with UID [-9193853349873940844]

Input message: test
Received response: TEST
test2
Input message: test2
Received response: TEST2

```

Output z aktora lokalnego

```

Z2_AppLocal x Z2_AppRemote x
[WARN] [05/11/2020 20:55:22.237] [main] [akka.remote.RemoteActorRefProvider] Using the 'remote' ActorRefProvider directly, which is a low-level layer. For most use cases, the 'cluster' abstraction on top of remoting is more suitable instead.
[WARN] [05/11/2020 20:55:22.238] [main] [akka.remote.RemoteActorRefProvider] Akka Cluster not in use - Using Akka Cluster is recommended if you need remote watch and deploy.
[INFO] [05/11/2020 20:55:22.707] [main] [ArteryTcpTransport(akka://remote_system)] Remoting started with transport [Artery tcp]; listening on address [akka://remote_system@127.0.0.1:2552] with UID [-902946844762475944]

Received message: test, Uppercased: TEST
Received message: test2, Uppercased: TEST2

```

Output z aktora zdalnego

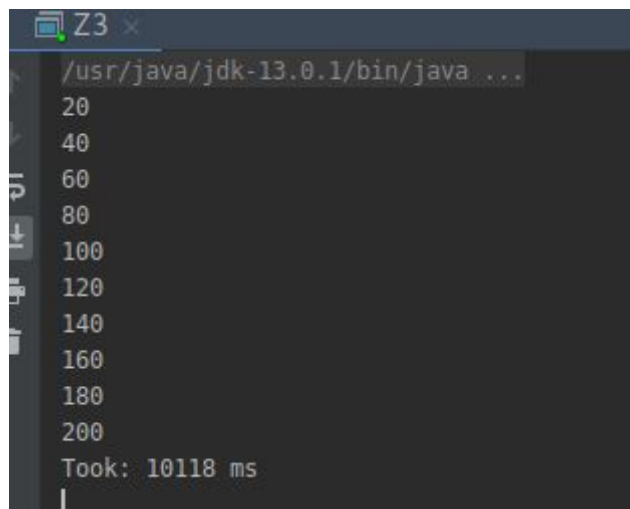
Wiadomości zostały poprawnie przesłane pomiędzy lokalnym, a zdalnym hostem. Aby rozróżnić wiadomość od odpowiedzi oraz przesłać path stworzone zostały dodatkowe klasy

Z2_Message oraz Z2_Response. Aby wesprzeć serializację stworzonych klas do konfigu należało dodać następującą konfigurację:

```
akka {
  actor {
    provider = remote
    allow-java-serialization = on
    serializers {
      java = "akka.serialization.JavaSerializer"
    }
    serialization-bindings {
      "java.lang.String" = java
      "Z2_Message" = java
      "Z2_Response" = java
    }
  }
  # provider = cluster
}
```

Konfiguracja wspierająca serializację obiektów

Zadanie 3.

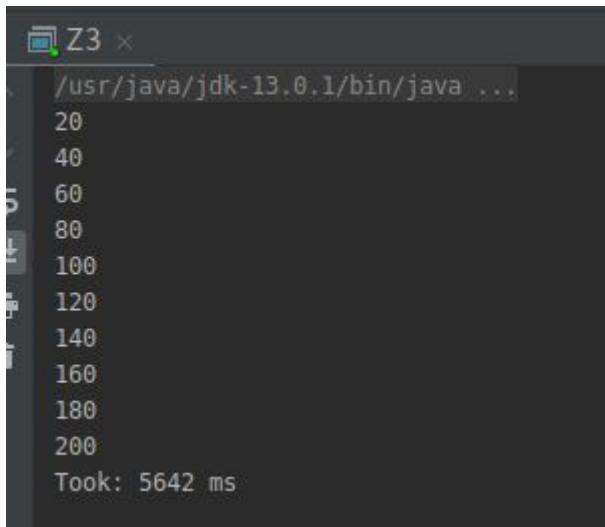


The screenshot shows a terminal window with a title bar containing a file icon and the text 'Z3 x'. The command prompt is `/usr/java/jdk-13.0.1/bin/java ...`. The output consists of a list of even numbers from 20 to 200, followed by the text 'Took: 10118 ms'.

```
/usr/java/jdk-13.0.1/bin/java ...
20
40
60
80
100
120
140
160
180
200
Took: 10118 ms
```

Bez async

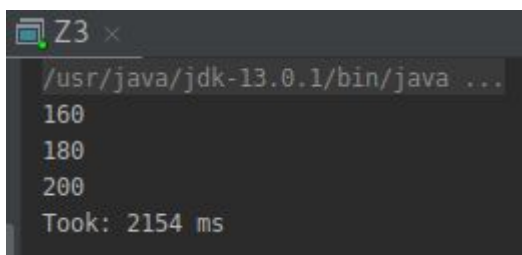
Początkowa konfiguracja wykonywała się ok. 10 sek co sugeruje, że operacje wykonały się sekwencyjnie (następna gdy zakończyła się poprzednia), dotyczy to zarówno kolejnych liczb z generatora w jednej operacji map oraz obu operacji map.



```
Z3 x
/usr/java/jdk-13.0.1/bin/java ...
20
40
60
80
100
120
140
160
180
200
Took: 5642 ms
```

Z async

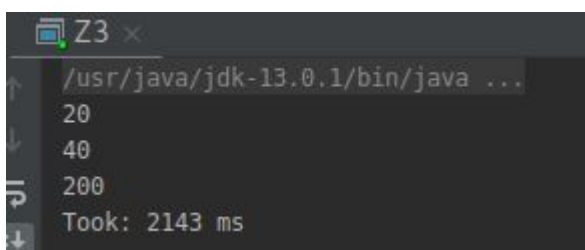
Po dodaniu opcji async czas zmalał do ok 5,5 sek. Sugeruje to że w ramach pojedynczej operacji map liczby nadal przetwarzane były sekwencyjnie, ale operacje obie map wykonały się asynchronicznie i gdy pierwsza wartość została przetworzona przez pierwszą operację map została ona od razu przekazana do drugiej operacji. Każda liczba musiała być przetworzona przez pierwszego map'a stąd dodatkowe pół sekundy.



```
Z3 x
/usr/java/jdk-13.0.1/bin/java ...
160
180
200
Took: 2154 ms
```

Z buffer dropHead

Bufor spowodował zmniejszenie ilości wyemitowanych wartości do 3 ostatnich. Gdy dochodziły kolejne liczby to zgodnie z strategią dropHead najstarsze były wyrzucone jeśli jeszcze nie były przetworzone, więc zostały tylko 3 ostatnie. Elementy wyparte nie były przetwarzane na co wskazuje niewielki czas wykonania całości (2 sek).

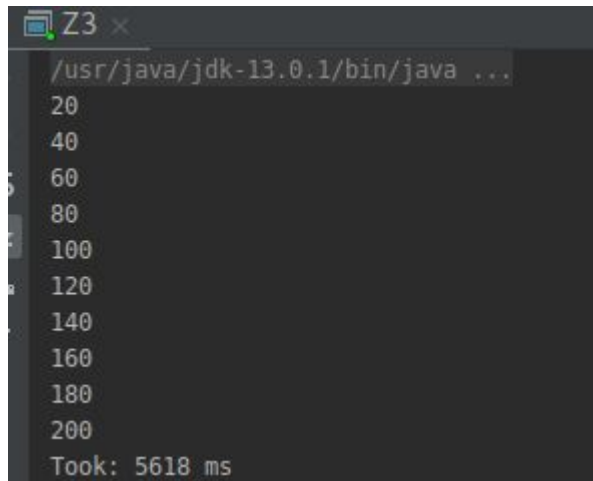


```
Z3 x
/usr/java/jdk-13.0.1/bin/java ...
20
40
200
Took: 2143 ms
```

Z buffer dropTail

Program zadziałał podobnie dla strategii dropTail z tą różnicą że kolejne dochodzące wartości wyrzucały te ostatnio dodane więc zostały dwie pierwsze (bo nigdy nie były ostatnie

w buforze) oraz ostatnia ponieważ nic już nie wyparło jej z bufora. Czas ponownie był mniejszy.

A terminal window titled 'Z3' with a close button. The command prompt shows the path to the Java binary: `/usr/java/jdk-13.0.1/bin/java ...`. The output consists of a series of even numbers from 20 to 200, with a final line indicating the execution time: `Took: 5618 ms`.

```
Z3 x
/usr/java/jdk-13.0.1/bin/java ...
20
40
60
80
100
120
140
160
180
200
Took: 5618 ms
```

Z buffer backpressure

Dla strategii backpressure wyemitowane zostały wszystkie wartości oraz nie zmienił się czas względem samego programu bez bufora. Dzieje się tak ponieważ backpressure sygnalizuje źródło, aby produkowało ono po prostu z mniejszą prędkością.