Course Authored By:

Srinivasan Kannan
(also known as: Shrinivaasan Kannan, Shrinivas Kannan)
Ph: 9791499106, 9003082186
Krishna iResearch Open Source Products Profiles:
http://sourceforge.net/users/ka_shrinivaasan,
https://github.com/shrinivaasanka,
https://www.openhub.net/accounts/ka_shrinivaasan
Personal website(research): https://sites.google.com/site/kuja27/
emails: ka.shrinivaasan@gmail.com, shrinivas.kannan@gmail.com,
kashrinivaasan@live.com

##
##

This is a non-linearly organized, code puzzles oriented, continually updated set of course notes on Java language. This complements NeuronRain course materials on Linux Kernel, Cloud, BigData Analytics and Machine Learning and covers fundamentals of Java.

----

7 February 2017

Generics in Java are equivalents of Standard Templates Library/Boost in C++ with facility to parametrize Classes and Function signatures. Java Generics define a variable, class, interface and function with syntax:

```
      variablename<T1,T2,...> v;
      classname<T1,T2,...> { };
      interfacename<T1,T2,...> { };
      T1 functionname(T2 t2,...);
```

Spark Streaming code in https://github.com/shrinivaasanka/asfer-github-code/blob/master/java-src/bigdata_analytics/spark_streaming/SparkGenericStreaming.java uses Generic JavaDStream with type parameter <String> and JavaPairDStream with parameter <String, Integer>. Diamond notation ignores typenames and runtime inference is done for following declaration:

```
      T1<T2,T3> t1 = new T1<>();
```

which automatically infers type as T1<T2,T3>.

21 May 2019

Lambda functions in Java are illustrated by LambdaExpressions.java which implements a functional interface and a lambda expression to print N consecutive numbers without looping. Function recursion() implements the recursive part while lambda expression just consists of printing an integer which is captured from argument to lambda function. Logs for this code example are in testlogs/LambdaExpressions.log.21May2019.