Chapter 1 : Scala Language

Exercises :

1. Learn about the Java language and its associated terms, such as JDK, JRE, bytecode, and JVM.
2. Learn why Scala was developed.
3. Research more about Apache Spark.
4. Understand the different use cases for Big Data.
5. Understand the various applications of Scala.
6. Research the other famous products developed in Scala, and if possible, why?
7. Research concerns about the performance of certain languages, such as Python

Answers :

1. Java is a modern programming language developed by Sun Microsystems (now acquired by Oracle). Java is a portable language because it can be deployed on several environments such as linux, windows, Mac. Java allows you to create window or console applications, applets, mobile applications, and many others.

   JDK (Java Development Kit) is a set of basic software libraries of the Java programming language, as well as the tools with which Java code can be compiled, transformed into bytecode intended for the Java virtual machine.

   JRE (Java Runtime Environment) is a Java development environment that consists of a virtual machine, software libraries used by Java programs and a plugin to allow the execution of these programs from web browsers

   Byte code: In computing, it is an intermediate code between the machine instructions and the source code, which is not directly executable.

   JVM (Java Virtual Machine) is a fictional computing device that runs programs compiled as Java bytecode. It executes the instructions given by the bytecode contained in the Java class following the stack model: each stack level contains the data specific to each operation.

2. Scala is a multi-paradigm programming language (object-oriented, imperative and functional paradigm) designed at the Swiss Federal Institute of Technology Lausanne (EPFL) to express common programming patterns in a concise and elegant form. Its name comes from the English Scalable language which roughly means “adaptable language”. It is intended to be compiled into Java bytecode (executable on the JVM), or .NET. Only the Java platform is officially supported by EPFL.

3. Spark is an open source distributed computing framework created by Martin Odersky in 2001. It is a set of tools and software components structured according to a defined architecture. Developed at the University of California at Berkeley by AMPLab, Spark is now a project of the Apache Foundation. Spark is now an Apache Foundation project. This

product is a big data processing application framework for performing large-scale complex analyses.

4. Spark is the new In-Memory brick of Hadoop distributions. Thanks to the richness of its libraries, Spark meets your Big Data needs or those requiring fast response times or to perform advanced calculations. The Spark solution interfaces with Yarn to benefit from the allocated resources. Spark integrates data analysis and data science tools. Indeed, Spark Streaming allows access to real-time data via the following tools: Spark SQL to query and modify data as with classic queries, Spark MLib for Machine Learning models and GraphX for calculation and creation of graphs.

5. Parallelization of tasks is made easier with Scala because many third-party libraries can be used for specific tasks. With fewer lines of code than Java, Scala takes less time to code. It also offers various tools and APIs that can be used for a wide variety of applications. In addition, Scala is used for a variety of use cases. It is used for writing web applications, for data streaming applications, for concurrent and distributed applications, for parallel batch processing, and for data analysis with Apache Spark.

6. Scala vs Python for Apache Spark

Apache Spark, the popular Big Data analytics framework, is written in Scala. This is what allows it to offer high speed due to its static nature. However, Spark offers APIs for Scala, Python, Java and R. The two most commonly used languages for Spark are Scala and Python.

In terms of performance, Scala is ten times faster than Python. Scala uses Java Virtual Machines during runtime, which provides increased speed in most cases. The dynamic nature of Python also reduces its speed.

Spark libraries must be called by Python, and this requires a lot of code processing. In this case, Scala works well with a limited number of cores.

In addition, Scala interacts better with Hadoop services and in particular the HDFS file system on which Spark is based. With Python, developers must use third-party libraries like Hadoopy, whereas Scala interacts with Hadoop via native Java APIs. This makes it easier to write native Hadoop applications in Scala.

Some data scientists prefer Scala and others prefer Python. The choice obviously depends on the use cases, but DataScientest recommends learning Python first.

Both languages are object-oriented and functional. Their syntax has similarities, and both have a large community of enthusiastic users. However, Scala can be a bit more difficult to learn than Python. However, it is better suited for more complex workflows. Python, on the other hand, has a simple syntax and many good libraries.

Thanks to its multiple libraries, Scala allows for the rapid integration of databases in Big Data ecosystems. The language allows writing code with multiple concurrency primitives, whereas Python does not support concurrency or multithreading. This concurrency feature allows Scala to provide better data processing and memory management.

Nevertheless, Python does support process forking. Only one thread is active at a time, and more processes must be restarted each time code is deployed. This increases the memory overhead.

In terms of usage, Scala and Python are two expressive languages that allow a high level of functionality. Python's strong point is its conciseness, and its more intuitive use. On the other hand, Scala is more powerful in terms of frameworks, libraries and macros. Its functional nature gives it a synergy with the Mapreduce framework.

Many Scala data frameworks follow abstract data types consistent with the Scala API collection. Developers need to learn the basic standard collections, and can then easily learn other libraries.

Note that Spark is written in Scala. Therefore, knowing Scala allows one to understand and modify the inner workings of Spark. In addition, many upcoming features will have APIs in Scala and Java first, and then in Python in later versions.

However, for Natural Language Processing (NLP), Python is preferred, as Scala does not offer many tools for Machine Learning and NLP. Similarly, Python is favored for the use of GraphX, GraphFrames, and MLLib. Python's visualization libraries complement Pyspark, and neither Spark nor Scala offer an equivalent.

Regarding security and code restoration, Scala is a static language that allows to find compile time errors. Python, on the other hand, is a dynamic language that is highly prone to bugs with each change made to the existing code. Refactoring code is therefore easier on Scala than on Python.

In conclusion, Python is slower and easier to use. Scala is faster and moderately easy to use. Since Spark is written in Scala, this language gives early access to new features. However, choosing the best language for Apache Spark depends on the needs of the project. While Python is more data analysis oriented, Scala is engineering oriented. However, both languages are excellent for building data science applications.