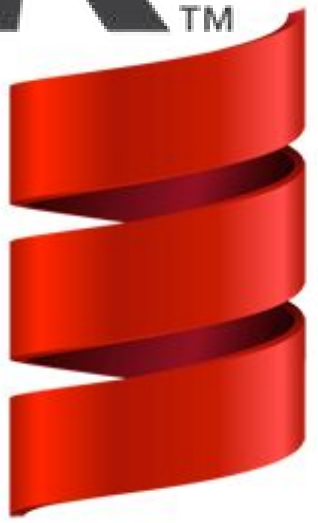
Spark Installation & Practicals



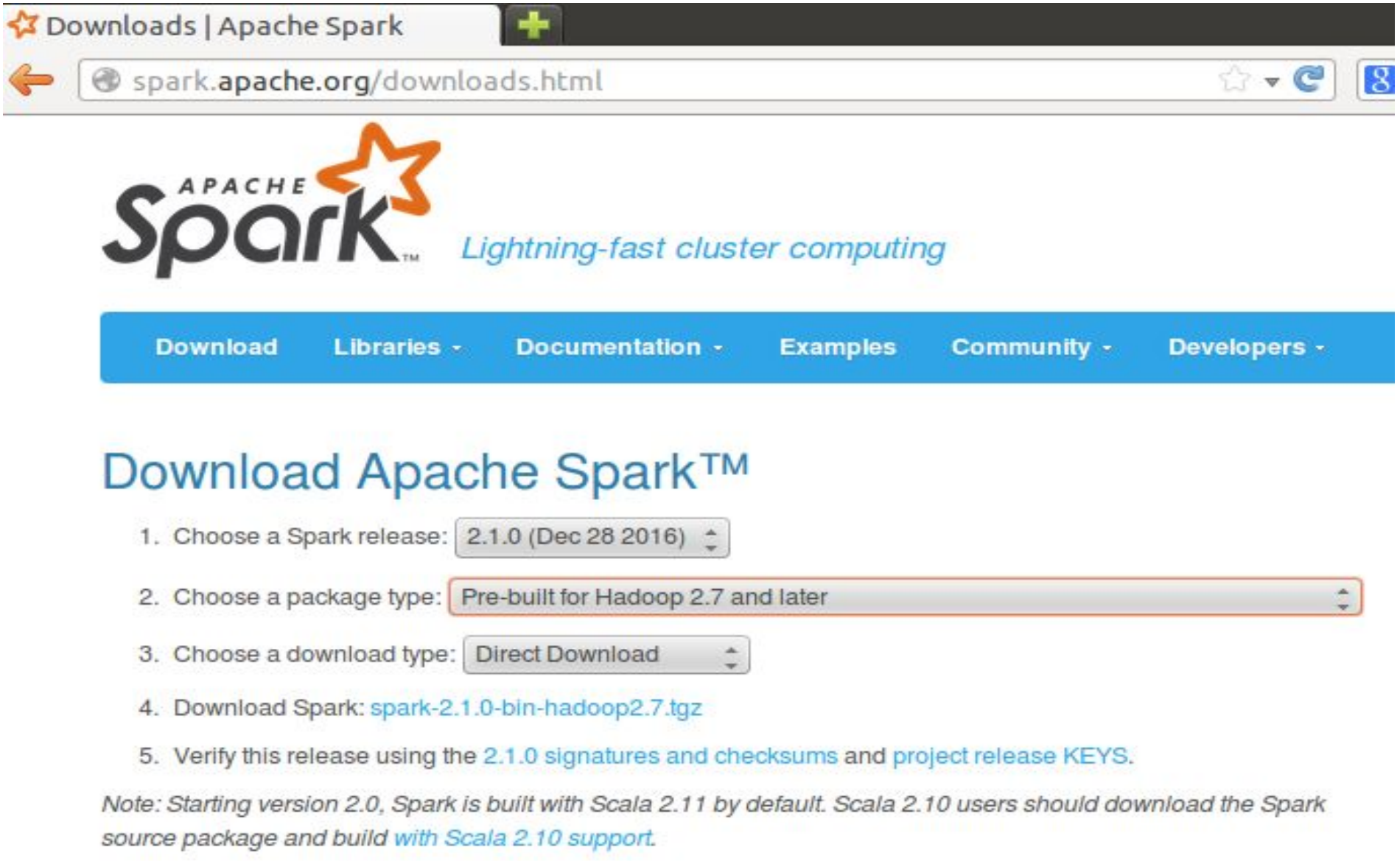
Scala

Agenda

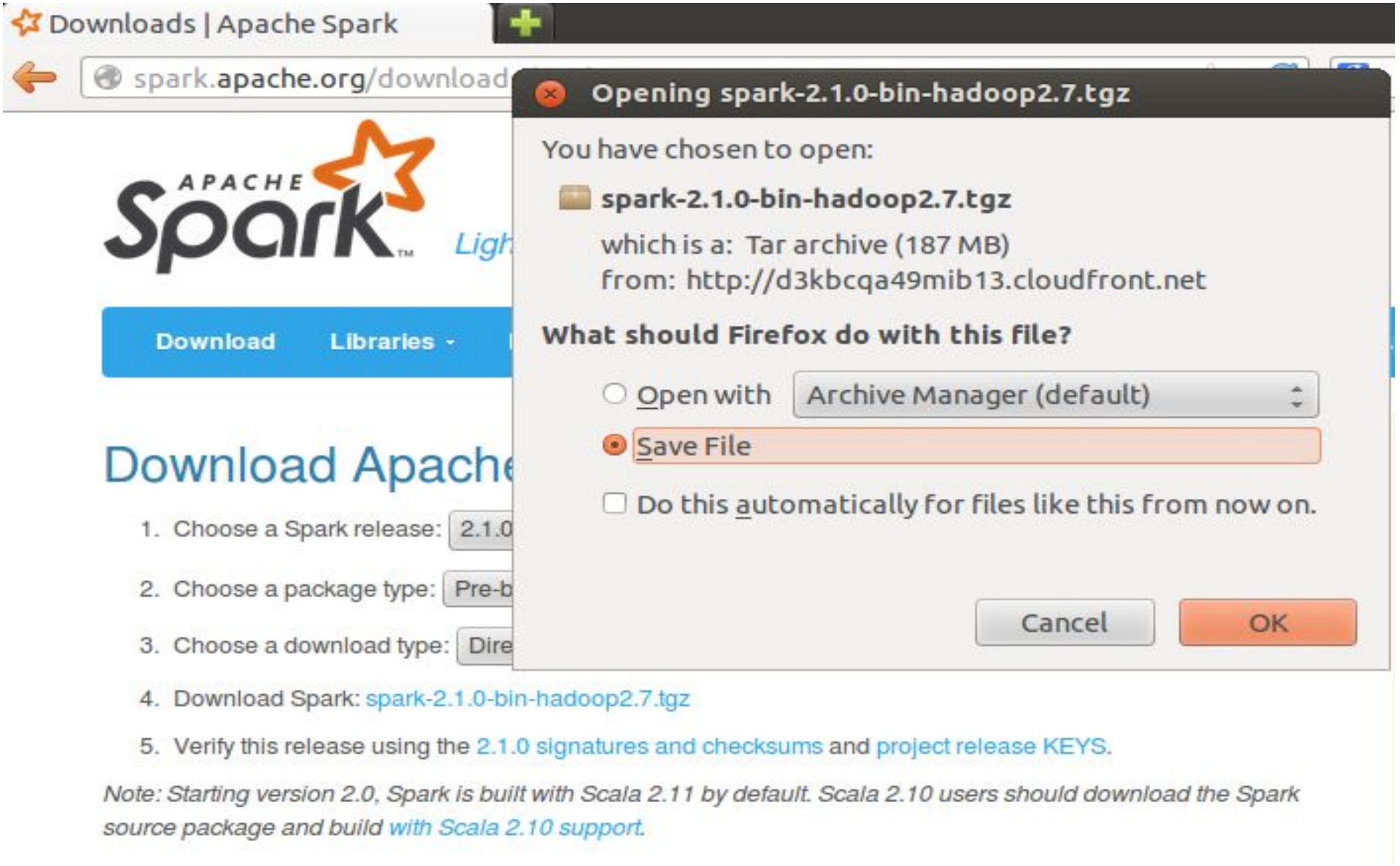
Visit Apache Spark website
Download Spark tar.gz file
Extract Spark tar.gz file
Open Spark Shell
Basic operation in Scala
constants & variables in Scala
If-Else in Scala
Use of braces in Scala
Paste mode in Scala
While loop in Scala
For loop in Scala
Function in Scala
Array in Scala
ArrayBuffer in Scala
Map in Scala
Word Count in Scala

Visit Apache Spark website

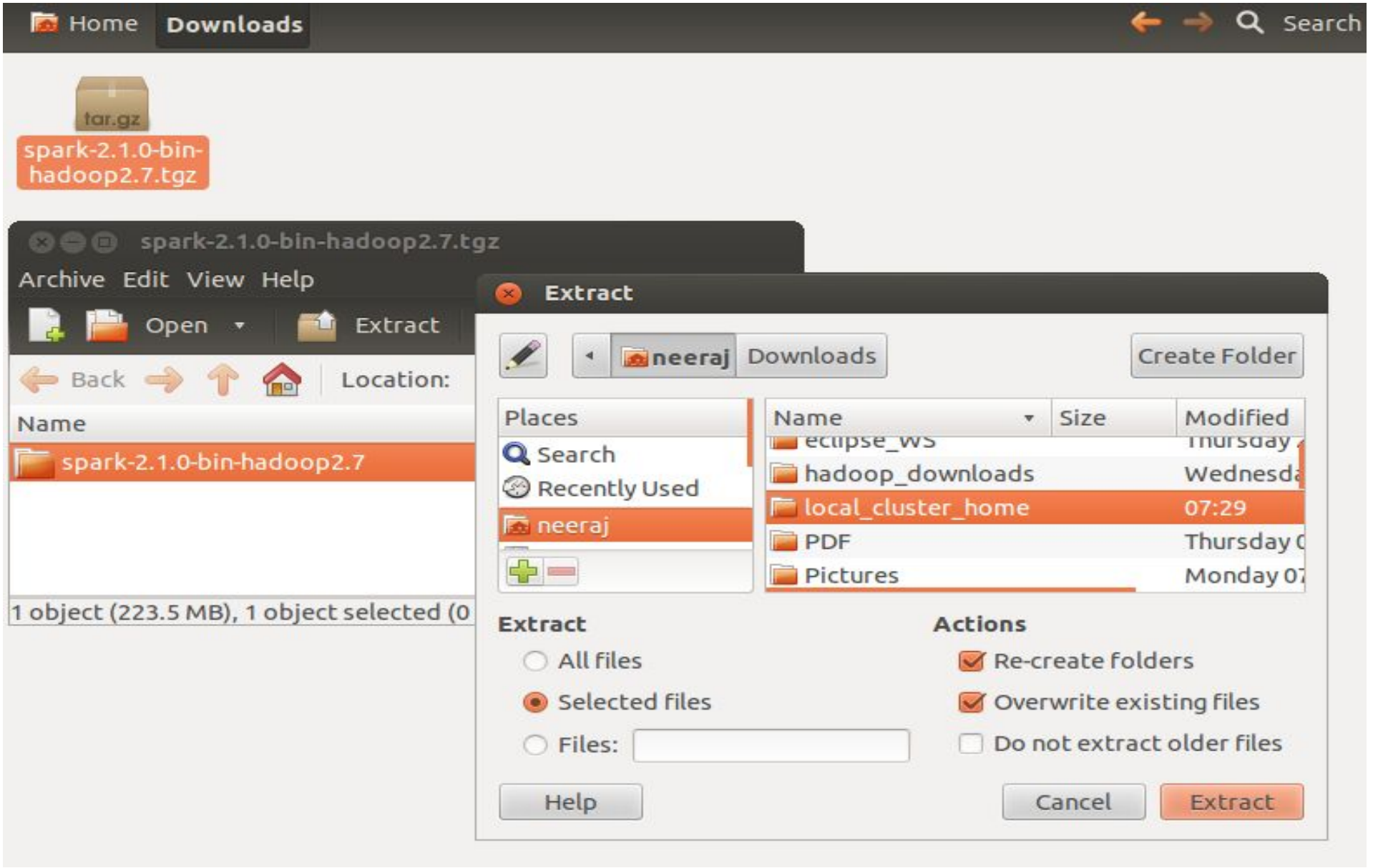
http://spark.apache.org/downloads.html



Download Spark tar.gz file



Extract Spark tar.gz file



Open Spark Shell

```
neeraj@myubuntu:~/local_cluster_home/spark-2.1.0-bin-hadoop2.7/bin$ pwd
/home/neeraj/local_cluster_home/spark-2.1.0-bin-hadoop2.7/bin
neeraj@myubuntu:~/local_cluster_home/spark-2.1.0-bin-hadoop2.7/bin$ ./spark-shell
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
17/01/17 22:24:37 WARN SparkContext: Support for Java 7 is deprecated as of Spark 2.0.0
17/01/17 22:24:38 WARN NativeCodeLoader: Unable to load native-hadoop library for your platf
uiltin-java classes where applicable
17/01/17 22:24:38 WARN Utils: Your hostname, myubuntu resolves to a loopback address: 127.0.
0.2.15 instead (on interface eth5)
17/01/17 22:24:38 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
17/01/17 22:25:03 WARN ObjectStore: Failed to get database global_temp, returning NoSuchObje
Spark context Web UI available at http://10.0.2.15:4040
Spark context available as 'sc' (master = local[*], app id = local-1484672079424).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.1.0
      /_/

Using Scala version 2.11.8 (Java HotSpot(TM) Client VM, Java 1.7.0_80)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

Basic operation in Scala

```
scala> 4 * 5 + 3
res0: Int = 23

scala> val result = 4 * 5 + 3
result: Int = 23
```

Constants & variables in Scala

```
scala> val answer = 5
answer: Int = 5

scala> answer = 8
<console>:25: error: reassignment to val
       answer = 8
              ^

scala> var answer = 5
answer: Int = 5

scala> answer = 8
answer: Int = 8
```

If-Else in Scala

```
scala> var a = 5
a: Int = 5

scala> var b = 0
b: Int = 0

scala> if (a > 0) b = 1 else b = -1

scala> b
res4: Int = 1

scala>
```

Use of braces in Scala

```
scala> var a = 2
a: Int = 2

scala> var b = 3
b: Int = 3

scala> if (a > 0) {
     | b = b * 2
     | b = b + 1
     | }
```

Paste mode in Scala

```
scala> :paste
// Entering paste mode (ctrl-D to finish)

var a = 2
var b = 3
if (a > 0 )
b = b * 2
b = b + 1
b

// Exiting paste mode, now interpreting.

a: Int = 2
b: Int = 7
b: Int = 7
res11: Int = 7
```

While loop in Scala

```
scala> var counter = 1
counter: Int = 1

scala> while(counter <= 5) {
     | println("Hello !!!")
     | counter = counter + 1
     | }
Hello !!!
Hello !!!
Hello !!!
Hello !!!
Hello !!!

scala>
```

For loop in Scala

```
scala> for(i <- 1 to 5) {
     | println("Hello" + i)
     | }
Hello1
Hello2
Hello3
Hello4
Hello5

scala>
```

For loop in Scala

```
scala> for(i <- 1 until 5) {
     | println("Hello" + i)
     | }
Hello1
Hello2
Hello3
Hello4

scala>
```

Function in Scala

```
scala> def myFxn(n : Int) = {
     |
     | var r = 1
     |
     | for (i <- 1 to n) r = r * i
     |
     | r
     |
     | }
myFxn: (n: Int)Int

scala> myFxn(5)
res18: Int = 120
```

Function in Scala

```
scala> def displayName(name: String,
     | left: String = "[ ",
     |  right: String = " ]") =
     | left + name + right
displayName: (name: String, left: String, right: String)

scala> displayName("SPARK")
res36: String = [ SPARK ]
```

Array in Scala

```
scala> val nums = new Array[Int](5)
nums: Array[Int] = Array(0, 0, 0, 0, 0)

scala> nums(0)=5

scala> nums(1)=7

scala> nums
res24: Array[Int] = Array(5, 7, 0, 0, 0)

scala> val words = Array("Hello", "World")
words: Array[String] = Array(Hello, World)

scala> words
res25: Array[String] = Array(Hello, World)

scala> words(0)="Hello!!!"

scala> words
res27: Array[String] = Array(Hello!!!, World)
```

Array in Scala

```
scala> var nums = Array(1,2,3,4,5)
nums: Array[Int] = Array(1, 2, 3, 4, 5)

scala> for (elem <- nums) yield 2 * elem
res28: Array[Int] = Array(2, 4, 6, 8, 10)

scala> var newNums = for (elem <- nums) yield 2 * elem
newNums: Array[Int] = Array(2, 4, 6, 8, 10)

scala> newNums
res29: Array[Int] = Array(2, 4, 6, 8, 10)
```

ArrayBuffer in Scala

```
scala> import scala.collection.mutable.ArrayBuffer
import scala.collection.mutable.ArrayBuffer

scala> val nums = ArrayBuffer(1, 7, 2, 9)
nums: scala.collection.mutable.ArrayBuffer[Int] = ArrayBuffer(1, 7, 2, 9)

scala> val sortedNums = nums.sorted
sortedNums: scala.collection.mutable.ArrayBuffer[Int] = ArrayBuffer(1, 2, 7, 9)

scala> val sortedNums = nums.sortWith(_ > _)
sortedNums: scala.collection.mutable.ArrayBuffer[Int] = ArrayBuffer(9, 7, 2, 1)
```

Map in Scala

```
scala> var map1 = Map("A" -> 1, "B" -> 2, "C" -> 3)
map1: scala.collection.immutable.Map[String,Int] = Map(A -> 1, B -> 2, C -> 3)

scala> map1("A")=0
<console>:27: error: value update is not a member of scala.collection.immutable.Map[String,Int]
       map1("A")=0
       ^

scala> var map2 = scala.collection.mutable.Map("A" -> 1, "B" -> 2, "C" -> 3)
map2: scala.collection.mutable.Map[String,Int] = Map(A -> 1, C -> 3, B -> 2)

scala> map2
res31: scala.collection.mutable.Map[String,Int] = Map(A -> 1, C -> 3, B -> 2)

scala> map2("A")=0

scala> map2
res33: scala.collection.mutable.Map[String,Int] = Map(A -> 0, C -> 3, B -> 2)
```

Word Count in Scala

```
scala> val inputPath = "/home/neeraj/sample_hadoop.txt"
inputPath: String = /home/neeraj/sample_hadoop.txt

scala> val outputPath = "/home/neeraj/Desktop/spark_word_count_op"
outputPath: String = /home/neeraj/Desktop/spark_word_count_op

scala> val input = sc.textFile(inputPath)
input: org.apache.spark.rdd.RDD[String] = /home/neeraj/sample_hadoop.txt MapPartitionsRDD[12] at textFile
at <console>:27

scala> val words = input.flatMap(line => line.split(" "))
words: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[13] at flatMap at <console>:29

scala> val count = words.map(word => (word, 1)).reduceByKey { case (x,y) => x + y}
count: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[15] at reduceByKey at <console>:33

scala> val output = count.map{case ((key, id)) => (key + " " + id)}
output: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[16] at map at <console>:35

scala> output.coalesce(1, true).saveAsTextFile(outputPath)
```

…Thanks…