第一次作业：

1. 请列举Hadoop生态系统的各个组件及其功能

HDFS：是Hadoop分布式文件系统，作用是将大量数据分布到计算机集群上，数据一次写入，可用于多次读取分析。

MapReduce：分布式计算框架，作用是对数据进行相关的处理，并将整理好的数据传输给HDFS。

Hive：作为数据仓库，可用SQL语句进行数据的操作，用于数据的存储。

Pig：作为数据流处理，用于分析Hadoop数据集的执行环境和脚本语言。

Mahout：数据挖掘和机器学习库，提供常见的算法。

HBase：面向列的NoSQL数据库，作用是对大量数据进行快速读取和写入。

Zookeeper：分布式协调服务，用于Hadoop操作的管理，是一个具有高度可用性的服务。

Flume：是一个分布式的，具有可靠性和高可用性的服务，用于收集，聚合和移动大量日志数据。

Sqoop：ETL工具，用于Hadoop与关系型数据库之间数据的移动。

Ambari：安装部署工具，用于对Hadoop集群进行供应、管理和监控的支持。

2.请阐述Hadoop生态系统中Hbase与其他部分的关系。

Hbase依赖于Zookeeper，默认情况下，它管理一个zookeeper实例。Hbase使用zookeeper作为分布式协调服务来维护集群中的服务器状态。Hbase中主节点Hmaster启动时将系统表加载到zookeeper集群上，通过zookeeper可以获取当前表的信息，两者之间互相通信，zookeeper作为Hbase的集群协调器，协调好Hbase的活动。HDFS为Hbase提供高可靠性的底层存储支持，MaoReduce为Hbase提供高性能的计算能力，Pig和Hive为Hbase提供高层语言支持，Sqoop和Hbase提供了方便的关系型数据库的数据的导入功能。

3.Hadoop主要有哪些缺点？相比之下，Spark具有哪些优点？

Hadoop缺点：延迟性高，对于IO操作的消耗大，不适合大量的小文件存储。

Spark优点：基于内存计算，延迟性较低，秒级响应，IO操作消耗比Hadoop较小，效率高，在迭代算法的效率和交互式分析（用户向数据集发出一系列专用的搜索性查询）的效率比HDFS要高。

4.简述Scala语言的基本特性

Scala是一种面向对象的、多范式的编程语言，每个值都是对象。同时也是一种函数式语言，其函数也能当作值来使用。Scala具备类型系统，通过编译时检查，保证代码的安全性和一致性。拥有良好的扩展性和并发性。

第二次作业：

1. 简述Scala语言和Java语言的联系的区别。

Scala吸收继承了多种语言中的优秀特性，它运行在Java虚拟机之上，轻松实现和丰富的Java类库互联互通。它既支持面向对象的编程方式，又支持函数式编程。Java与Scala语言都是基于JVM的语言。区别在于Scala语言时不支持接口interface，采用trait，而Java默认时protected。Scala是面向对象和面向函数式的编程语言，而Java是面向对象的语言。

1. Scala有哪些基本数据类型和操作符？

数据类型：Byte、Short、Int、Long、Float、Double、Char、String、Boolean、Unit、Null、Nothing、Any、AnyRef。

操作符：

算术操作符：+ - \* / %

关系运算符：== != > < >= <=

逻辑运算符：&& || !

等。

1. 请分别用脚本和编译运行两种形式输出“Hello World”。

object Test{

def main(args:Array[String]){

println(“Hello World”)

}

}

scala> println(“Hello World”)

第三次作业：

1. Scala是如何实现Java循环中的break和continue功能的？

Scala要实现break的话，就必须创建一个Breaks对象，用法如下:

val a = new Breaks; //创建Breaks对象

1. breakable{

for(…){

a.break; //中断循环

}

}

Scala中continue的另一种用法，在scala中没有continue这个关键字，用法如下：

for(…){

breakable{

if(…){

break()

}

}

}

1. 什么是单例对象和伴生对象？

单例对象与伴生对象的区别在于是否有一个类与对象的名是否相同，例子如下：

object Cat{…} //在这里只定义了一个名为Cat对象

class Board(…){…} //在这里定义了一个名为Board对象

object Board{…} //在这里定义了一个名为Board对象

在上面可以看出，Cat为单例对象，Board为伴生对象，因为Board有其相同名的类。

1. 什么是case类，它和普通类的区别是什么？

case class为样例类，是一种特殊的类，常用于模式匹配。

可以不使用new关键字就可以构建对象。以下的例子可以看出case class的用法：

object Test { def main(args: Array[String]) {

val alice = new Person("Alice", 25)

val bob = new Person("Bob", 32)

val charlie = new Person("Charlie", 32)

for (person <- List(alice, bob, charlie))

{ person match {

case Person("Alice", 25) => println("Hi Alice!")

case Person("Bob", 32) => println("Hi Bob!")

case Person(name, age) => println("Age: " + age + " year, name: " + name + "?")

} } }

// 样例类

case class Person(name: String, age: Int) }

输出结果为：

Hi Alice!

Hi Bob!

Age: 32 year, name: Charlie?