

# 课程实验二：实践 MapReduce 分布式数据处理

## 一、实验描述

本实验使用 IDEA 构建大数据工程，通过 Java 语言编写 WordCount 程序并通过集群运行，完成单词计数任务。首先，在本地进行 IDEA 的安装，接着使用 IDEA 构建大数据工程并编写 Wordcount 程序，最后将程序打包在先前实验构建的集群上运行程序。

## 二、实验目的

1. 了解 IDEA 构建大数据工程的过程；
2. 熟悉使用 Java 语言编写大数据程序；
3. 了解 MapReduce 的工作原理；
4. 掌握在集群上运行程序的方法。

## 三、实验环境

1. 系统版本：Centos 7.5；
2. Hadoop 版本：Apache Hadoop 2.7.7；
3. JDK 版本：1.8.\*；
4. IDEA 版本：IDEA2021.3。

## 四、实验步骤

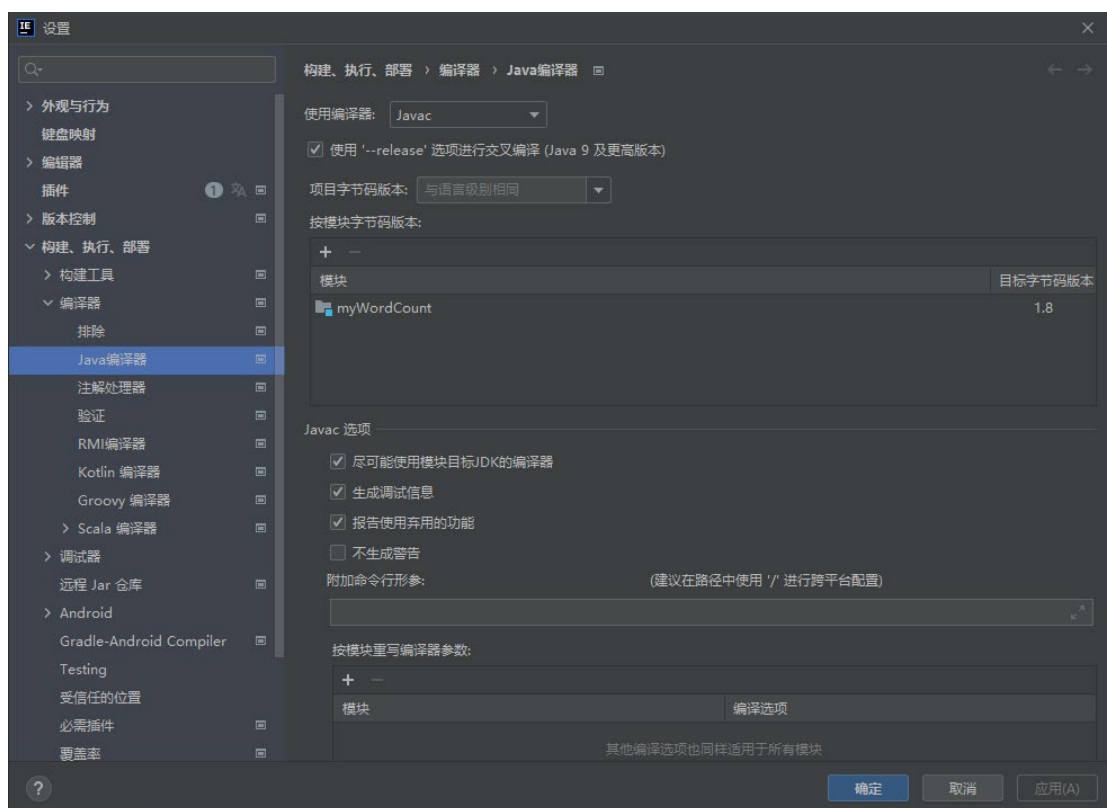
### 4.1 IDEA 构建大数据工程

步骤 1：创建项目，打开 IDEA，创建工程

步骤 2：依赖设置

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4       xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5
6       <modelVersion>4.0.0</modelVersion>
7
8       <groupId>org.example</groupId>
9       <artifactId>myWordCount</artifactId>
10      <version>1.0-SNAPSHOT</version>
11
12      <properties>
13          <maven.compiler.source>16</maven.compiler.source>
14          <maven.compiler.target>16</maven.compiler.target>
15          <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
16          <hadoop.version>2.7.7</hadoop.version>
17      </properties>
18
19      <dependencies>
20          <dependency>
21              <groupId>org.apache.hadoop</groupId>
22              <artifactId>hadoop-client</artifactId>
23              <version>${hadoop.version}</version>
24          </dependency>
25          <dependency>
26              <groupId>org.apache.hadoop</groupId>
27              <artifactId>hadoop-common</artifactId>
28              <version>${hadoop.version}</version>
29          </dependency>
30          <dependency>
31              <groupId>org.apache.hadoop</groupId>
32              <artifactId>hadoop-hdfs</artifactId>
33              <version>${hadoop.version}</version>
34          </dependency>
35      </dependencies>
36
37 </project>
```

### 步骤 3：设置语言环境



#### 步骤 4：设置 java Compiler 环境

1) 点击菜单栏中的 file，选择 Setting

2) 依次选择 Build，Execution—>Compiler—>Java Compiler，设置 Project bytecode version 为 1.8，设置图中的 Target bytecode version 为 1.8，然后依次点击 Apply 和 OK；

## 4.2 WordCount 程序编写

- 1) 依次打开 src—>main—>java，在 java 上点击右键，创建 Java Class；
- 2) 输入类名 WordCount，点击 ok
- 3) 在类 WordCount 中添加 TokenizerMapper 类，并在该类中实现 map 函数；map 函数负责统计输入文件中单词的数量
- 4) 在类 WordCount 中添加 IntSumReducer 类，并在该类中实现 reduce 函数；reduce 函数合并之前 map 函数统计的结果，并输出最终结果；
- 5) 在类 WordCount 中添加 main 方法；
- 6) 创建 Configuration 对象，运行 MapReduce 程序前都要初始化。

Configuration，该类主要是读取 MapReduce 系统配置信息；

7) 限定输出参数必须为 2 个

8) 创建 Job 对象,第一行构建一个 job，构建时候有两个参数，一个是 conf，一个是这个 job 的名称。

9)定义输出的 key/value 的类型，也就是最终存储在 HDFS 上结果文件的 key/value 的类型。

10) 构建输入的数据文件和输出的数据文件，如果 job 运行成功了，程序就会正常退出。

```
// 2019211279-rcx
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;

import java.io.IOException;
import java.util.StringTokenizer;

public class WordCount {
    //该类中实现 map 函数
    public static class TokenizerMapper extends Mapper<Object,
Text,Text, IntWritable> {
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();
        //map 函数负责统计输入文件中单词的数量
        public void map(Object key,Text value,Context context) throws
IOException, InterruptedException{
            StringTokenizer itr = new
StringTokenizer(value.toString());
            while (itr.hasMoreTokens()){
                word.set(itr.nextToken());
                context.write(word,one);
            }
        }
    }
}
```

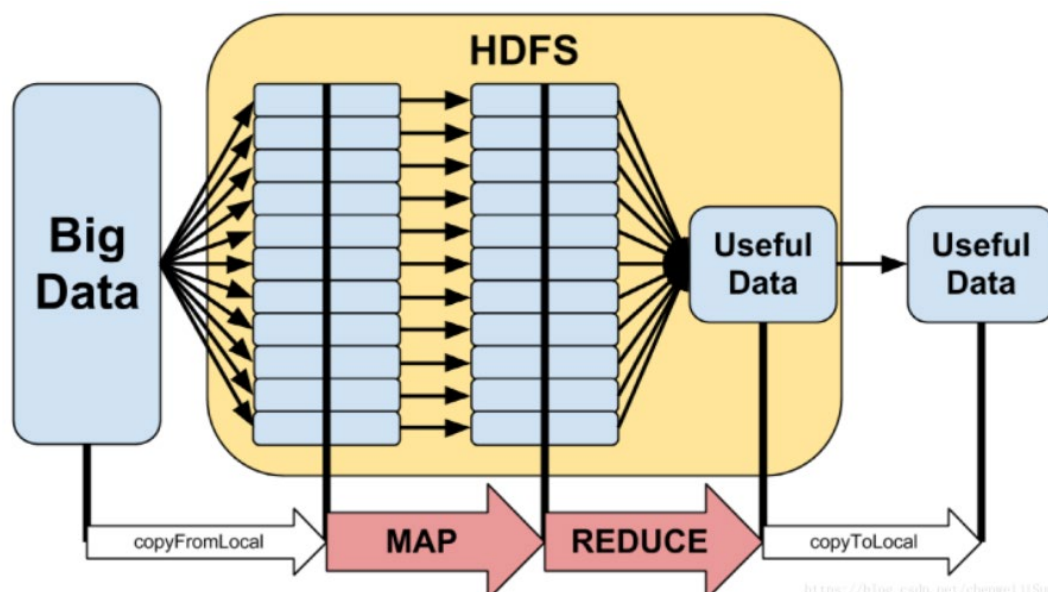
```

    }
    //该类中实现 reduce 函数
    public static class IntSumReducer extends
Reducer<Text,IntWritable,Text,IntWritable>{
        private IntWritable result = new IntWritable();
        //reduce 函数合并之前 map 函数统计的结果
        public void reduce(Text key, Iterable<IntWritable>
values,Context context) throws IOException,InterruptedException{
            int sum=0;
            for(IntWritable val :values){
                sum+=val.get();
            }
            result.set(sum);
            context.write(key,result);
        }
    }

    public static void main(String[] args) throws Exception{
        //创建 Configuration 对象，运行 MapReduce 程序前都要初始化
        Configuration，该类主要是读取 MapReduce 系统配置信息
        Configuration conf = new Configuration();
        //限定输出参数必须为 2 个
        String[] otherArgs = new GenericOptionsParser(conf,
args).getRemainingArgs();
        //运行 WordCount 程序时候一定是两个参数，如果不是就会输出错误提示并
        退出：
        if(otherArgs.length!=2){
            System.err.println("Usage:wordcount <in> <out>");
            System.exit(2);
        }
        //创建 Job 对象
        Job job =new Job(conf,"word count");
        //装载编写好的计算程序
        job.setJarByClass(WordCount.class);
        //载 Map 函数和 Reduce 函数实现类
        job.setMapperClass(TokenizerMapper.class);
        job.setCombinerClass(IntSumReducer.class);
        job.setReducerClass(IntSumReducer.class);
        //定义输出的 key/value 的类型
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        //构建输入的数据文件
        FileInputFormat.addInputPath(job,new Path(otherArgs[0]));
        //构建输出的数据文件

```

```
FileOutputFormat.setOutputPath(job,new Path(otherArgs[1]));  
//如果 job 运行成功了，程序就会正常退出  
System.exit(job.waitForCompletion(true)? 0:1);  
}  
}
```



#### 4.3 程序打包与运行

步骤 1: 打开 File->Project Structure:

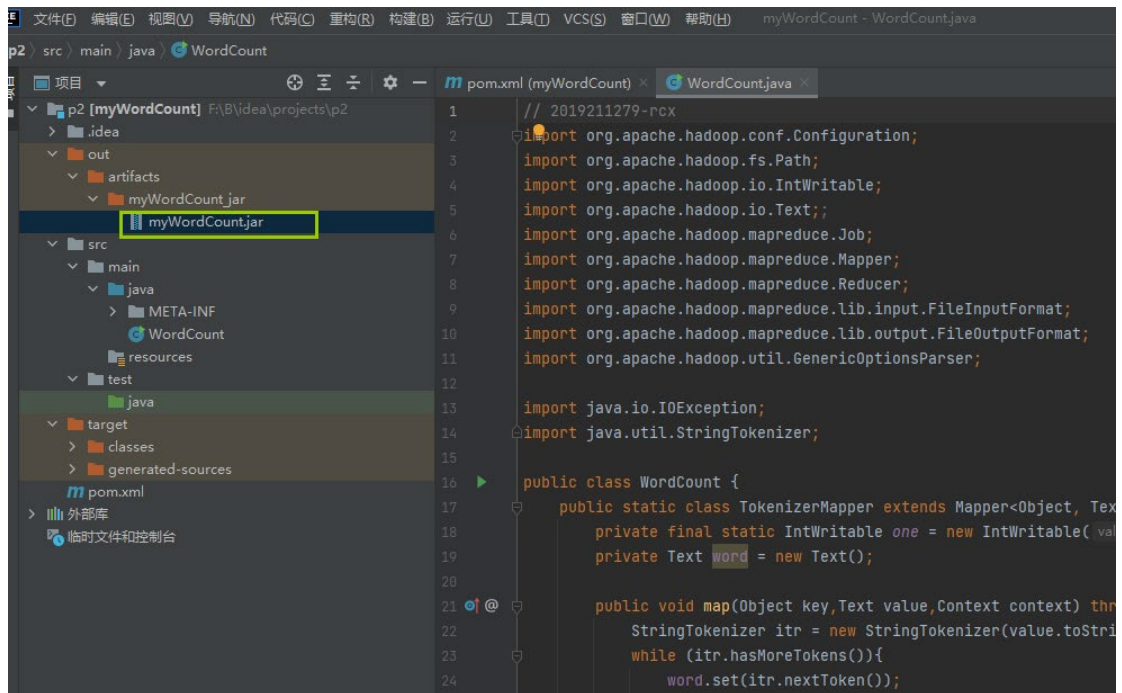
步骤 2: Project Settings 栏下的 Artifacts, 点击“+”,选择 JAR->From

modules with dependencies...:

步骤 3: 填写主类名称:

步骤 4: 选择 Build->Artifacts:

步骤 5: 选择 Build: 建立完成后会在 out 文件夹下生成 jar 包:

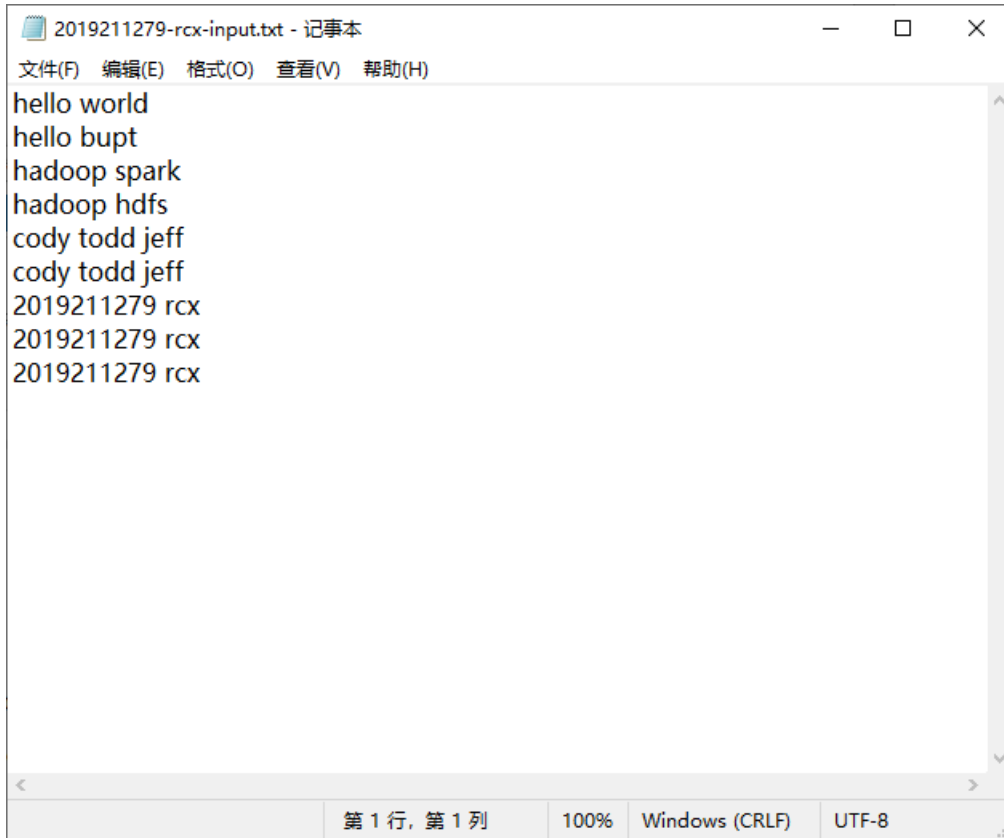


步骤 6: 使用压缩软件打开生成的 jar 包 (可以去本地电脑里找):

步骤 7: 找到 META-INF 目录, 并删除 MANIFEST.MF 文件

步骤 8: 使用 WinSCP 上传处理后的 jar 包到服务器:

步骤 9: 构建输入文件, 可在自己电脑构建然后上传到服务器中。格式为 学号-姓名简写-input.txt, 可通过 cat 命令打印检查。



步骤 10: 使用“`hadoop jar <jar 包名> <主函数> <其余参数>`”命令，在 hadoop 运行程序：

```
root@rcx-2019211279-0001 ~]# hadoop fs -ls -R /
22/04/05 02:40:29 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-
va classes where applicable
drwxr-xr-x - root supergroup          0 2022-04-05 02:22 /output
-rw-r--r--  3 root supergroup        61 2022-03-25 17:06 /rcx_2019211279.txt
-rw-r--r--  3 root supergroup       107 2022-03-25 17:06 /upload_2019211279.txt
drwxr-xr-x - root supergroup          0 2022-04-05 02:03 /user
drwxr-xr-x - root supergroup          0 2022-04-05 02:03 /user/hdfs
drwxr-xr-x - root supergroup          0 2022-04-05 02:39 /wordcount
-rw-r--r--  3 root supergroup       130 2022-04-05 02:39 /wordcount/2019211279-rcx-input.txt
root@rcx-2019211279-0001 ~]# hadoop jar myWordCount.jar WordCount /wordcount/2019211279-rcx-input.txt /output/rcx-201
1279-out.txt
```

得到如下图运行结果：



```
cat 选择root@rcx-2019211279-0001:~  
HDFS: Number of write operations=2  
Job Counters  
  Launched map tasks=1  
  Launched reduce tasks=1  
  Data-local map tasks=1  
  Total time spent by all maps in occupied slots (ms)=6179  
  Total time spent by all reduces in occupied slots (ms)=6596  
  Total time spent by all map tasks (ms)=6179  
  Total time spent by all reduce tasks (ms)=6596  
  Total vcore-milliseconds taken by all map tasks=6179  
  Total vcore-milliseconds taken by all reduce tasks=6596  
  Total megabyte-milliseconds taken by all map tasks=6327296  
  Total megabyte-milliseconds taken by all reduce tasks=6754304  
Map-Reduce Framework  
  Map input records=9  
  Map output records=20  
  Map output bytes=203  
  Map output materialized bytes=137  
  Input split bytes=131  
  Combine input records=20  
  Combine output records=11  
  Reduce input groups=11  
  Reduce shuffle bytes=137  
  Reduce input records=11  
  Reduce output records=11  
  Spilled Records=22  
  Shuffled Maps =1  
  Failed Shuffles=0  
  Merged Map outputs=1  
  GC time elapsed (ms)=154  
  CPU time spent (ms)=1500  
  Physical memory (bytes) snapshot=361693184  
  Virtual memory (bytes) snapshot=2568028160  
  Total committed heap usage (bytes)=171245568  
Shuffle Errors  
  BAD_ID=0  
  CONNECTION=0  
  IO_ERROR=0  
  WRONG_LENGTH=0  
  WRONG_MAP=0  
  WRONG_REDUCE=0  
File Input Format Counters  
  Bytes Read=130  
File Output Format Counters  
  Bytes Written=87  
[root@rcx-2019211279-0001 ~]#
```

步骤 11: 在 hdfs 上查看程序的输出:

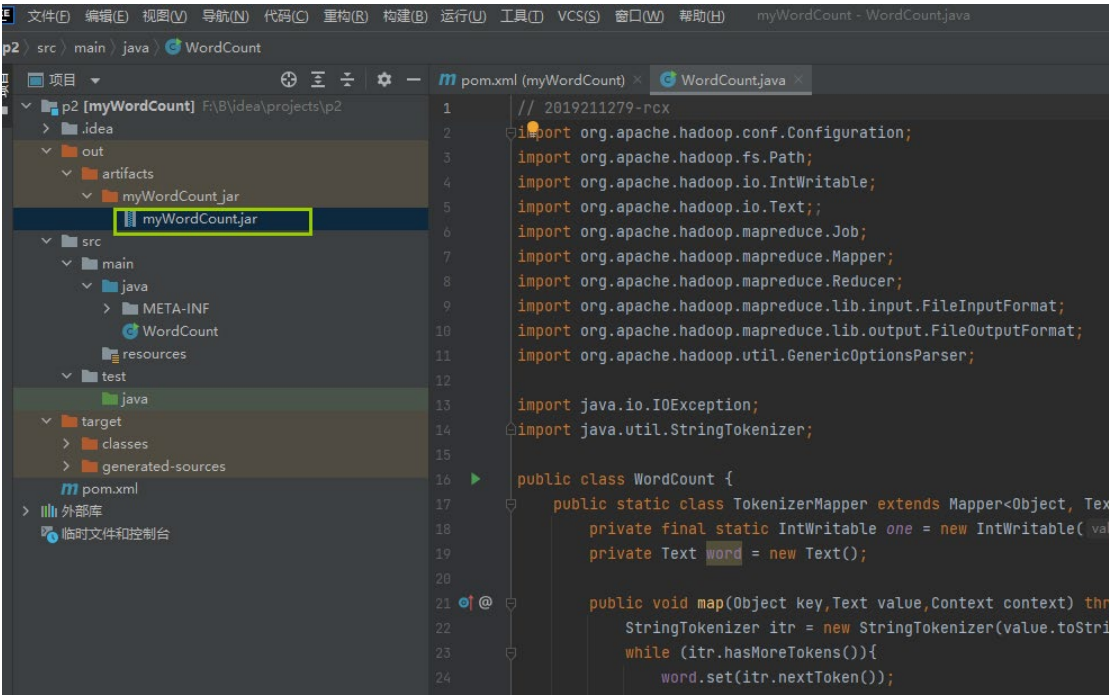
```

root@rcx-2019211279-0001:~
drwx----- - root supergroup      0 2022-04-05 02:43 /tmp/hadoop-yarn/staging
drwxr-xr-x - root supergroup      0 2022-04-05 02:43 /tmp/hadoop-yarn/staging/history
drwxrwxrwt - root supergroup      0 2022-04-05 02:43 /tmp/hadoop-yarn/staging/history/done_intermediate
drwxrwx--- - root supergroup      0 2022-04-05 02:43 /tmp/hadoop-yarn/staging/history/done_intermediate/root
-rwxrwx--- 3 root supergroup      33701 2022-04-05 02:43 /tmp/hadoop-yarn/staging/history/done_intermediate/root/job_1
649094707753_0001-1649097805374-root-word+count-1649097834016-1-1-SUCCEEDED-root.root-1649097815909.jhist
-rwxrwx--- 3 root supergroup      347 2022-04-05 02:43 /tmp/hadoop-yarn/staging/history/done_intermediate/root/job_1
649094707753_0001.summary
-rwxrwx--- 3 root supergroup      123847 2022-04-05 02:43 /tmp/hadoop-yarn/staging/history/done_intermediate/root/job_1
649094707753_0001.conf.xml
drwx----- - root supergroup      0 2022-04-05 02:43 /tmp/hadoop-yarn/staging/root
drwx----- - root supergroup      0 2022-04-05 02:43 /tmp/hadoop-yarn/staging/root/.staging
drwxrwxrwt - root root            0 2022-04-05 02:43 /tmp/logs
drwxrwx--- - root root            0 2022-04-05 02:43 /tmp/logs/root
drwxrwx--- - root root            0 2022-04-05 02:43 /tmp/logs/root/logs
drwxrwx--- - root root            0 2022-04-05 02:44 /tmp/logs/root/logs/application_1649094707753_0001
-rw-r----- 3 root root            8070 2022-04-05 02:44 /tmp/logs/root/logs/application_1649094707753_0001/rcx-201921
1279-0002_42175
-rw-r----- 3 root root            39831 2022-04-05 02:44 /tmp/logs/root/logs/application_1649094707753_0001/rcx-201921
1279-0003_42263
-rw-r----- 3 root root            5383 2022-04-05 02:44 /tmp/logs/root/logs/application_1649094707753_0001/rcx-201921
1279-0004_38507
-rw-r--r-- 3 root supergroup       107 2022-03-25 17:06 /upload_2019211279.txt
drwxr-xr-x - root supergroup      0 2022-04-05 02:03 /user
drwxr-xr-x - root supergroup      0 2022-04-05 02:03 /user/hdfs
drwxr-xr-x - root supergroup      0 2022-04-05 02:39 /wordcount
-rw-r--r-- 3 root supergroup       130 2022-04-05 02:39 /wordcount/2019211279-rcx-input.txt
[root@rcx-2019211279-0001 ~]# hadoop fs -cat /output/rcx-2019211279-out.txt/part-r-0000
cat: '/output/rcx-2019211279-out.txt/part-r-0000': No such file or directory
[root@rcx-2019211279-0001 ~]# hadoop fs -cat /output/rcx-2019211279-out.txt/part-r-00000
22/04/05 02:50:57 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-j
va classes where applicable
2019211279      3
bupt      1
cody      2
hadoop    2
hdfs      1
hello     2
jeff      2
rcx       3
spark     1
todd      2
world     1
[root@rcx-2019211279-0001 ~]#

```

# 五、实验结果与分析

## (1) 粘贴实验结果图 28

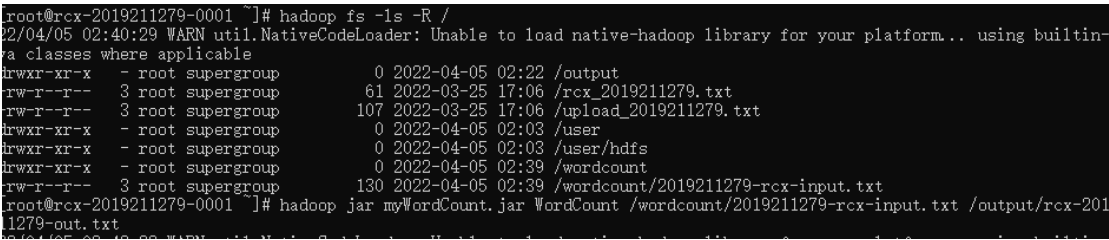


The screenshot shows an IDE with the following components:

- Project Structure (Left):** A tree view showing the project 'p2 [myWordCount]' with subdirectories 'src', 'target', and 'generated-sources'. The 'src/main/java' directory contains the 'WordCount' package, which includes the 'WordCount.java' file. The 'myWordCount.jar' file is highlighted in the 'artifacts' section.
- Code Editor (Right):** The 'WordCount.java' file is open, showing the following code:

```
1 // 2019211279-rcx
2 import org.apache.hadoop.conf.Configuration;
3 import org.apache.hadoop.fs.Path;
4 import org.apache.hadoop.io.IntWritable;
5 import org.apache.hadoop.io.Text;
6 import org.apache.hadoop.mapreduce.Job;
7 import org.apache.hadoop.mapreduce.Mapper;
8 import org.apache.hadoop.mapreduce.Reducer;
9 import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
10 import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
11 import org.apache.hadoop.util.GenericOptionsParser;
12
13 import java.io.IOException;
14 import java.util.StringTokenizer;
15
16 public class WordCount {
17     public static class TokenizerMapper extends Mapper<Object, Text> {
18         private final static IntWritable one = new IntWritable(1);
19         private Text word = new Text();
20
21         public void map(Object key, Text value, Context context) throws IOException, InterruptedException {
22             StringTokenizer itr = new StringTokenizer(value.toString());
23             while (itr.hasMoreTokens()) {
24                 word.set(itr.nextToken());
```

## (2) 粘贴实验结果图 33



The screenshot shows a terminal window with the following command and output:

```
root@rcx-2019211279-0001 ~]# hadoop fs -ls -R /
22/04/05 02:40:29 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
drwxr-xr-x - root supergroup          0 2022-04-05 02:22 /output
-rw-r--r--  3 root supergroup      61 2022-03-25 17:06 /rcx_2019211279.txt
-rw-r--r--  3 root supergroup    107 2022-03-25 17:06 /upload_2019211279.txt
drwxr-xr-x - root supergroup          0 2022-04-05 02:03 /user
drwxr-xr-x - root supergroup          0 2022-04-05 02:03 /user/hdfs
drwxr-xr-x - root supergroup          0 2022-04-05 02:39 /wordcount
-rw-r--r--  3 root supergroup    130 2022-04-05 02:39 /wordcount/2019211279-rcx-input.txt
root@rcx-2019211279-0001 ~]# hadoop jar myWordCount.jar WordCount /wordcount/2019211279-rcx-input.txt /output/rcx-2019211279-out.txt
```

## (3) 粘贴实验结果图 34

```
cat 选择root@rcx-2019211279-0001:~
HDFS: Number of write operations=2
Job Counters
  Launched map tasks=1
  Launched reduce tasks=1
  Data-local map tasks=1
  Total time spent by all maps in occupied slots (ms)=6179
  Total time spent by all reduces in occupied slots (ms)=6596
  Total time spent by all map tasks (ms)=6179
  Total time spent by all reduce tasks (ms)=6596
  Total vcore-milliseconds taken by all map tasks=6179
  Total vcore-milliseconds taken by all reduce tasks=6596
  Total megabyte-milliseconds taken by all map tasks=6327296
  Total megabyte-milliseconds taken by all reduce tasks=6754304
Map-Reduce Framework
  Map input records=9
  Map output records=20
  Map output bytes=203
  Map output materialized bytes=137
  Input split bytes=131
  Combine input records=20
  Combine output records=11
  Reduce input groups=11
  Reduce shuffle bytes=137
  Reduce input records=11
  Reduce output records=11
  Spilled Records=22
  Shuffled Maps =1
  Failed Shuffles=0
  Merged Map outputs=1
  GC time elapsed (ms)=154
  CPU time spent (ms)=1500
  Physical memory (bytes) snapshot=361693184
  Virtual memory (bytes) snapshot=2568028160
  Total committed heap usage (bytes)=171245568
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=130
File Output Format Counters
  Bytes Written=87
[root@rcx-2019211279-0001 ~]#
```

(4) 粘贴实验结果图 35

```
root@rcx-2019211279-0001:~  
drwx----- - root supergroup 0 2022-04-05 02:43 /tmp/hadoop-yarn/staging  
drwxr-xr-x - root supergroup 0 2022-04-05 02:43 /tmp/hadoop-yarn/staging/history  
drwxrwxrwt - root supergroup 0 2022-04-05 02:43 /tmp/hadoop-yarn/staging/history/done_intermediate  
drwxrwx--- - root supergroup 0 2022-04-05 02:43 /tmp/hadoop-yarn/staging/history/done_intermediate/root  
-rwxrwx--- 3 root supergroup 33701 2022-04-05 02:43 /tmp/hadoop-yarn/staging/history/done_intermediate/root/job_1  
649094707753_0001-1649097805374-root-word+count-1649097834016-1-1-SUCCEEDED-root.root-1649097815909.jhist  
-rwxrwx--- 3 root supergroup 347 2022-04-05 02:43 /tmp/hadoop-yarn/staging/history/done_intermediate/root/job_1  
649094707753_0001.summary  
-rwxrwx--- 3 root supergroup 123847 2022-04-05 02:43 /tmp/hadoop-yarn/staging/history/done_intermediate/root/job_1  
649094707753_0001.conf.xml  
drwx----- - root supergroup 0 2022-04-05 02:43 /tmp/hadoop-yarn/staging/root  
drwx----- - root supergroup 0 2022-04-05 02:43 /tmp/hadoop-yarn/staging/root/.staging  
drwxrwxrwt - root root 0 2022-04-05 02:43 /tmp/logs  
drwxrwx--- - root root 0 2022-04-05 02:43 /tmp/logs/root  
drwxrwx--- - root root 0 2022-04-05 02:43 /tmp/logs/root/logs  
drwxrwx--- - root root 0 2022-04-05 02:44 /tmp/logs/root/logs/application_1649094707753_0001  
-rw-r----- 3 root root 8070 2022-04-05 02:44 /tmp/logs/root/logs/application_1649094707753_0001/rcx-201921  
1279-0002_42175  
-rw-r----- 3 root root 39831 2022-04-05 02:44 /tmp/logs/root/logs/application_1649094707753_0001/rcx-201921  
1279-0003_42263  
-rw-r----- 3 root root 5383 2022-04-05 02:44 /tmp/logs/root/logs/application_1649094707753_0001/rcx-201921  
1279-0004_38507  
-rw-r--r-- 3 root supergroup 107 2022-03-25 17:06 /upload_2019211279.txt  
drwxr-xr-x - root supergroup 0 2022-04-05 02:03 /user  
drwxr-xr-x - root supergroup 0 2022-04-05 02:03 /user/hdfs  
drwxr-xr-x - root supergroup 0 2022-04-05 02:39 /wordcount  
-rw-r--r-- 3 root supergroup 130 2022-04-05 02:39 /wordcount/2019211279-rcx-input.txt  
[root@rcx-2019211279-0001 ~]# hadoop fs -cat /output/rcx-2019211279-out.txt/part-r-0000  
22/04/05 02:50:57 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-ja  
va classes where applicable  
cat: '/output/rcx-2019211279-out.txt/part-r-0000': No such file or directory  
[root@rcx-2019211279-0001 ~]# hadoop fs -cat /output/rcx-2019211279-out.txt/part-r-00000  
22/04/05 02:51:38 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-ja  
va classes where applicable  
2019211279 3  
bupt 1  
cody 2  
hadoop 2  
hdfs 1  
hello 2  
jeff 2  
rcx 3  
spark 1  
todd 2  
world 1  
[root@rcx-2019211279-0001 ~]#
```

(5) 提交 jar 包

见压缩包

(6) 解释 wordcount 程序代码，3 个类的作用，类之间的关系

见实验步骤中代码注释

(7) 实验中应对各个截图进行简单解释，证明理解截图含义

见实验步骤