Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Ордена Трудового Красного знамени

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Московский технический университет связи и информатики»

Кафедра Математической кибернетики и информационных технологий

ЛАБОРАТОРНАЯ РАБОТА № 3

по дисциплине

БОЛЬШИЕ ДАННЫЕ

Выполнил студент группы

БСТ1903

Козлов В. А

Вариант № 7

Москва 2021г.

СОДЕРЖАНИЕ

[ЦЕЛЬ РАБОТЫ 3](#_Toc88742350)

[1 ХОД РАБОТЫ 4](#_Toc88742351)

[1.1 Создание папок и текстовых файлов 4](#_Toc88742352)

[1.1 Код программы 4](#_Toc88742353)

[1.3 Компиляция класса сборка .jar файла 7](#_Toc88742354)

[1.4 Запуск программы WordCount. 7](#_Toc88742355)

# ЦЕЛЬ РАБОТЫ

Ознакомиться с процессом MapReduce на примере подсчета слов в файле.

# 1 ХОД РАБОТЫ

## 1.1 Создание папок и текстовых файлов

Создание папок и текстовых файлов в качестве образцов представлено на рисунке 1

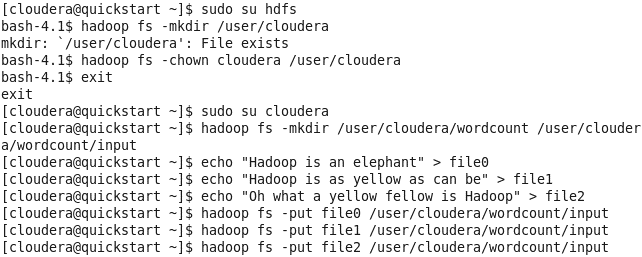


Рисунок 1 – Создание папки и файлов-образцов для программы

## 1.1 Код программы

import java.io.IOException;

import java.util.StringTokenizer;

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;

public class WordCount {

public static class TokenizerMapper

extends Mapper<Object, Text, Text, IntWritable>{

private final static IntWritable one = new IntWritable(1);

private Text word = new Text();

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);

}

}

}

public static class IntSumReducer

extends Reducer<Text,IntWritable,Text,IntWritable> {

private IntWritable result = new IntWritable();

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 {

Configuration conf = new Configuration();

String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();

if (otherArgs.length < 2) {

System.err.println("Usage: wordcount <in> [<in>...] <out>");

System.exit(2);

}

Job job = Job.getInstance(conf, "word count");

job.setJarByClass(WordCount.class);

job.setMapperClass(TokenizerMapper.class);

job.setCombinerClass(IntSumReducer.class);

job.setReducerClass(IntSumReducer.class);

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);

for (int i = 0; i < otherArgs.length - 1; ++i) {

FileInputFormat.addInputPath(job, new Path(otherArgs[i]));

}

FileOutputFormat.setOutputPath(job,

new Path(otherArgs[otherArgs.length - 1]));

System.exit(job.waitForCompletion(true) ? 0 : 1);

}

}

## 1.3 Компиляция класса сборка .jar файла

Компиляция класса и сборка .jar файла представлена на рисунке 2.

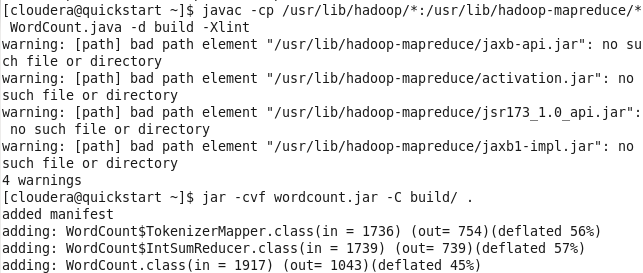


Рисунок 2 – Компиляция класса и сборка .jar файла

## 1.4 Запуск программы WordCount.

Результат выполнения представлен на рисунках 3–5.

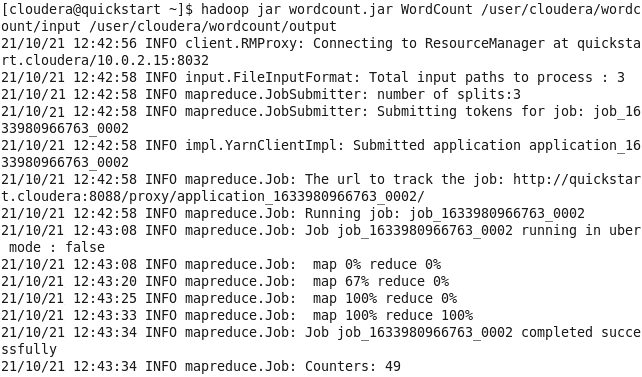


Рисунок 3 – Выполнение программы

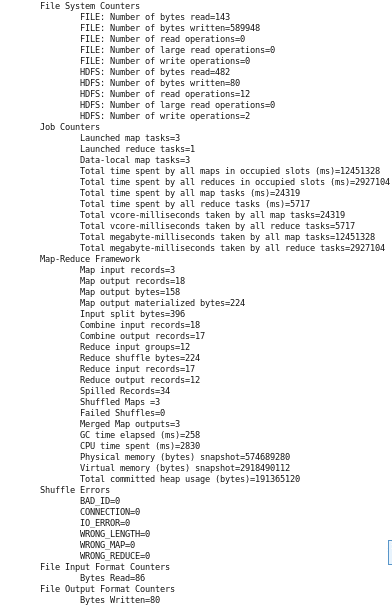


Рисунок 4 – Различные counter-ы (статистика выполнения)

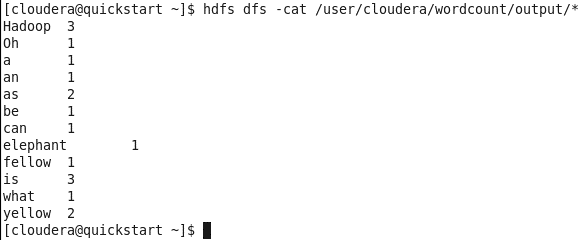


Рисунок 5 – Результат выполнения в файле