EXPERIMENT - 4

Aim:

Develop a MapReduce program to calculate the average marks of students and assign grades based on input data.

Theory:

In educational data processing, we often need to aggregate marks from different subjects for a specific student to determine their final standing. Using MapReduce, the 'Map' phase handles the ingestion of student records (Name, Subject, Marks) and organizes them by the student's name (Key). The 'Reduce' phase receives all marks associated with a specific student, calculates the average, and applies conditional logic to assign a grade (e.g., A, B, C, or Fail).

Description:

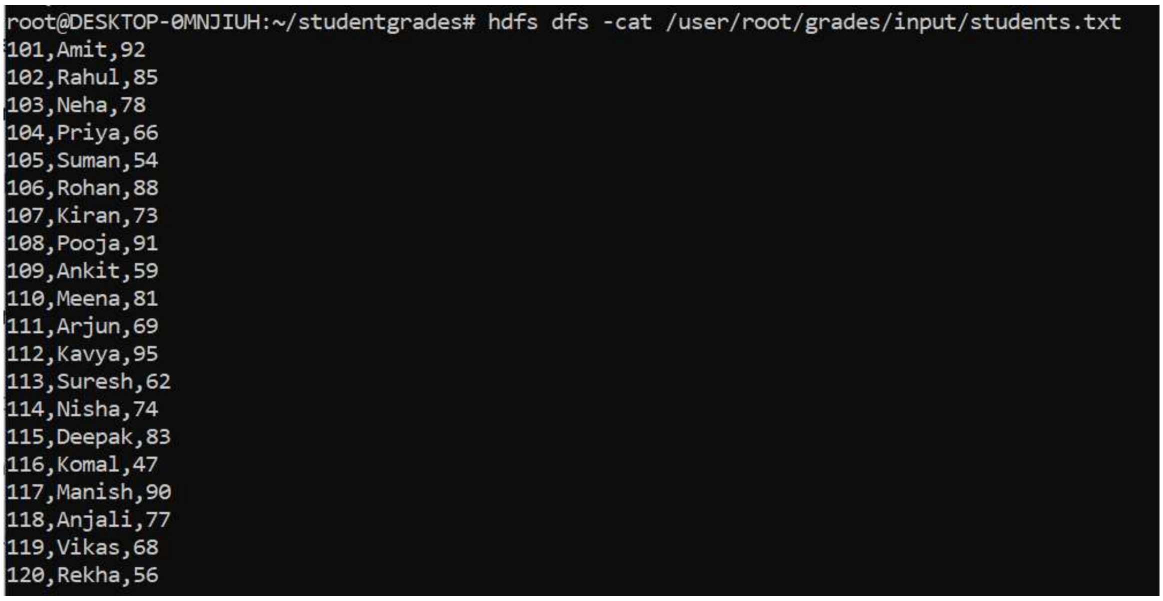
The program reads a text file where each line contains 'StudentName Subject Marks'. The Mapper emits the StudentName as the key and the Marks as the value. The Reducer iterates through the list of marks for each student, sums them up, counts the number of subjects, and calculates the average. Based on the average, a grade is assigned (Average >= 90: A, >= 80: B, else: C) and the final output is written.

Code:

```
import java.io.IOException;
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;

public class StudentGrade {

    // 1. Mapper Class
    public static class GradeMapper extends Mapper<Object, Text, Text,
IntWritable> {

        private Text studentName = new Text();
        private IntWritable marks = new IntWritable();

        public void map(Object key, Text value, Context context) throws
IOException, InterruptedException {
            // Input format: Name Subject Marks (e.g., "John Math 85")
            String[] data = value.toString().split("\s+");
            if (data.length >= 3) {
                studentName.set(data[0]);
                marks.set(Integer.parseInt(data[2]));
                context.write(studentName, marks);
            }
        }
    }

    // 2. Reducer Class
    public static class GradeReducer extends Reducer<Text, IntWritable, Text, 
Text> {

        public void reduce(Text key, Iterable<IntWritable> values, Context 
context) throws IOException, InterruptedException {
            int sum = 0;
            int count = 0;

            for (IntWritable val : values) {
                sum += val.get();
                count++;
            }

            double average = (double) sum / count;
            String grade;

            if (average >= 90) {
                grade = "A";
            } else if (average >= 80) {
                grade = "B";
            } else if (average >= 60) {
                grade = "C";
            } else {
                grade = "Fail";
            }

            context.write(key, new Text(grade));
        }
    }

    // 3. Driver / Main Method
    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "student grade");

        job.setJarByClass(StudentGrade.class);
        job.setMapperClass(GradeMapper.class);
        job.setReducerClass(GradeReducer.class);

        // Mapper outputs Text, IntWritable
        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(IntWritable.class);

        // Reducer outputs Text, Text (Name, Grade)
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

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

Input:

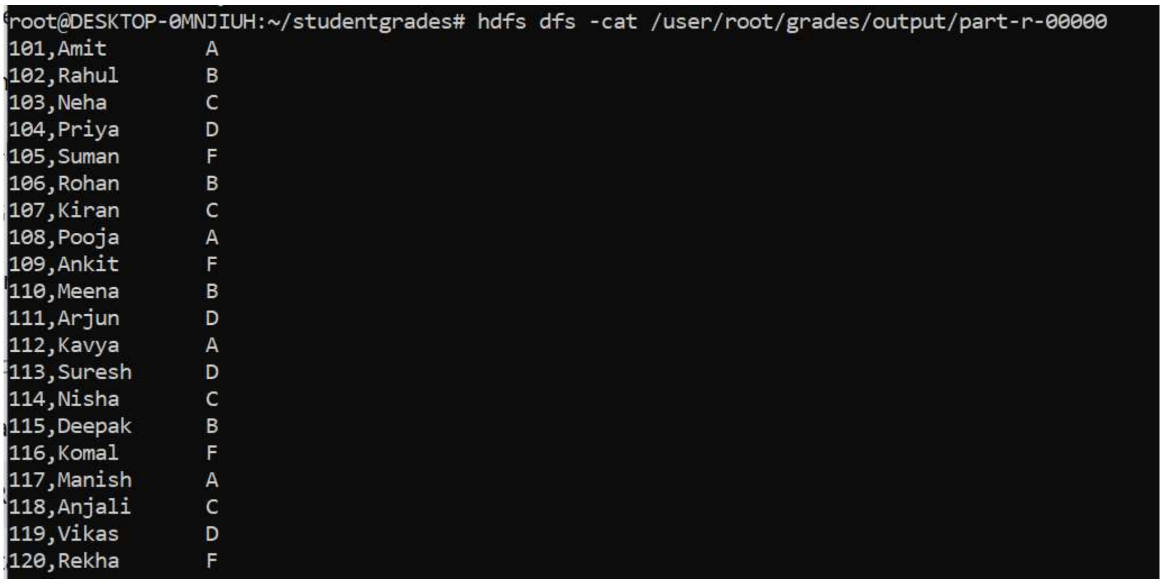
A text file named marks.txt containing the following text:



```
root@DESKTOP-0MNJIUH:~/studentgrades# hdfs dfs -cat /user/root/grades/input/students.txt
101,Amit,92
102,Rahul,85
103,Neha,78
104,Priya,66
105,Suman,54
106,Rohan,88
107,Kiran,73
108,Pooja,91
109,Ankit,59
110,Meena,81
111,Arjun,69
112,Kavya,95
113,Suresh,62
114,Nisha,74
115,Deepak,83
116,Komal,47
117,Manish,90
118,Anjali,77
119,Vikas,68
120,Rekha,56
```

Output:

The content of the output file generated by Hadoop:



```
root@DESKTOP-0MNJIUH:~/studentgrades# hdfs dfs -cat /user/root/grades/output/part-r-00000
101,Amit        A
102,Rahul       B
103,Neha        C
104,Priya       D
105,Suman       F
106,Rohan       B
107,Kiran       C
108,Pooja       A
109,Ankit       F
110,Meena       B
111,Arjun       D
112,Kavya       A
113,Suresh      D
114,Nisha       C
115,Deepak      B
116,Komal       F
117,Manish      A
118,Anjali      C
119,Vikas       D
120,Rekha       F
```

Commands:

Description	Command
Create input directory in HDFS:	hdfs dfs -mkdir /grade_input
Upload the local text file to HDFS:	hdfs dfs -put marks.txt /grade_input
Compile the Java program:	javac -classpath $(hadoop classpath) -d . StudentGrade.java
Create the JAR file:	jar -cvf grades.jar *.class
Run the Hadoop Job:	hadoop jar grades.jar StudentGrade /grade_input /grade_output
View the Output:	hdfs dfs -cat /grade_output/part-r-00000

Conclusion:

In this experiment, we successfully implemented a MapReduce program to process student marks. We learned how to aggregate data based on a specific key (Student Name) and how to implement conditional logic within the Reducer to determine the final grade based on calculated averages.