Big Data Management and Analytics
Assignment 8
(a) Compare and highlight the differences between Spark and Flink

<table>
<thead>
<tr>
<th>Features</th>
<th>Apache Flink</th>
<th>Apache Spark</th>
</tr>
</thead>
<tbody>
<tr>
<td>Streaming engine</td>
<td>Stream approach: A batch is a finite set of streamed data</td>
<td>Micro-batch approach: A stream is 'cut' into small batches</td>
</tr>
<tr>
<td>Iterative processing</td>
<td>Native iteration support</td>
<td>Non-native iteration, implemented as regular for-loops outside the system</td>
</tr>
<tr>
<td>Latency</td>
<td>Low latency, high throughput</td>
<td>High latency compared to Flink</td>
</tr>
<tr>
<td>Time management</td>
<td>Out-of-order events, windows, user-defined</td>
<td>Process time-based</td>
</tr>
</tbody>
</table>
(b) In how far is Flink more suitable for streaming tasks?

- provides natively a stream-processing approach
- has a lower latency
- supports more powerful windowing systems
- has explicit time-handling
Building blocks of an Apache Flink program:

FlatMap functions take elements and transform them into zero, one or more elements in a non-nested structure.

```
import org.apache.flink.api.common.functions.FlatMapFunction;
import org.apache.flink.api.java.DataSet;
import org.apache.flink.api.java.ExecutionEnvironment;
import org.apache.flink.api.java.tuple.Tuple2;
import org.apache.flink.util.Collector;
```

- Represents a collection of elements of the same type.
- Context in which program is executed.
- Tuples have a fixed length and contain a set of fields which can be of different types.
- Collects a record and forwards it.
Building blocks of an Apache Flink program:

Create a context object in which the program is executed

Create a DataSet of strings by reading out the text file

```java
public class FlinkProgram {
    public static void main(String[] args) throws Exception {
        ExecutionEnvironment env = ExecutionEnvironment.getExecutionEnvironment();
        DataSet<String> rawdata = env.readTextFile("C:\\Users\\kazempour\\Documents\\ttwist.txt");
        DataSet<Tuple2<String, Integer>> result = rawdata
            .flatMap(new Splitter())
            .groupBy(0)
            .sum(1);
        result.print();
    }
}
```

Create a resulting DataSet consisting of 2-tuples by...

...mapping a splitter method

...summing up (reducing) the number of their occurrences

...grouping the resulting tuples according to their words
Building blocks of an Apache Flink program:

```java
public static class Splitter implements FlatMapFunction<String, Tuple2<String, Integer>> {
    @Override
    public void flatMap(String line, Collector<Tuple2<String, Integer>> out) {
        for (String wordToken : line.split(" ")) {
            out.collect(new Tuple2<String, Integer>(wordToken, 1));
        }
    }
}
```

Method takes a string and a collector as a 2-tuple and appends a collection filled with 2-tuples of the structure: (wordToken, #ofOccurrence)
Assignment 8-3

See Java-Code!