Big Data Analytics with Scala and Apache Spark

M.Sc. course on "Technologies for Big Data Analysis" - Assignment 3

CHRISTOS BALAKTSIS (506) and VASILEIOS PAPASTERGIOS (505), Aristotle University, Greece

1 Introduction

The current document is a technical report for the third programming assignment in the M.Sc. course on *Technologies for Big Data Analysis*, offered by the *DWS M.Sc Program*[1] of the Aristotle University of Thessaloniki, Greece. The course is taught by Professor Apostolos Papadopoulos [2]. The authors attended the course during their first year of Ph.D. studies at the Institution.

The assignment contains 4 sub-problems and is part of a series, comprising 3 programming assignments on the following topics:

Assignment 1 Multi-threading Programming and Inter-Process Communication
Assignment 2 The Map-Reduce Programming Paradigm
Assignment 3 Big Data Analytics with Scala and Apache Spark

In this document we focus on Assignment 3 and its 4 sub-problems. We refer to them as *problems* in the rest of the document for simplicity. The source code of our solution has been made available at the following public repository in the GitHub platform: `https://github.com/Bilpapster/big-data-playground`.

Roadmap. The rest of our work is structured as follows. We devote one section to each one of the 4 problems. That means problems 1, 2, 3 and 4 are presented in section 2, section 3, section 4 and section 5 respectively. For each problem, we first provide the problem statement, as given by the assignment. Next, we thoroughly present the reasoning and/or methodology we have adopted to approach the problem and devise a solution. Wherever applicable, we also provide insights about the source code implementation we have developed. Finally, we conclude our work in section 6. The appendix includes the evaluation results for any issues that necessitated them.

2 Problem 1: Numeronyms

We discuss here the first problem of the assignment. The main target of the assignment is to get familiar with a simple task leveraging Apache Spark and Scala programming language. This is a WordCount problem's variation.

2.1 Problem Statement

Implement an **Apache Spark** (Scala) program, a variation of the **word-count** problem, to compute the average length of words that start with a specific letter (a-z). The program should sort the results based on the average length, printing first the letters with higher average length. For example:

k − 8.2
a − 5.6
b − 4.8

---

[1]https://dws.csd.auth.gr/
[2]https://datalab-old.csd.auth.gr/~apostol/

Authors' Contact Information: Christos Balaktsis (506), balaktsis@csd.auth.gr; Vasileios Papastergios (505), papster@csd.auth.gr, Aristotle University, Thessaloniki, Greece.

Note that you may opt for preprocessing the input text, e.g., transforming all letters to lowercase or ignoring words that start with a number.

2.2 Proposed approach

2.2.1 Setting. Our implementation is run and tested in a Linux environment with 12 cores, using the Scala programming language version 2.13.15 and Apache Spark version 3.5.3. We have used SBT as the build tool of our solution. We have used the Software Development Kit (JDK) version 11.0.11. The source code is developed in IntelliJ IDEA Community Edition 2021.1.1 and managed using SBT as the build tool. All dependencies of the project can be found in the `build.sbt` file located at the root folder of the project. The project is compiled and executed directly from the IntelliJ IDEA.

To run the project, open the `src/main/scala/Task1WordLength.scala` file in IntelliJ IDEA, and execute the main method. After successful execution of the program, the results can be viewed under the `output/task1` directory.

2.2.2 Implementation. The proposed approach leverages the Apache Spark computing framework to calculate the average word length for each one of the 26 letters in the Latin alphabet. In particular, the input text is read and tokenized into words. As part of a preprocessing step, all letters are transformed to lowercase and all words that start with a number are dropped.

In order to compute the average length for each initial letter, we leverage the RDD API of Apache Spark. More specifically, we use a `PairRDD` that stores composite key-value pairs in the form (`letter, (word_length, 1)`). The key of each pair is the initial letter of the respective word (`letter`). The value of each pair is a pair itself, having as (false) key the length of the word (`word_length`) and as (false) value a unary value (`1`). We use the term *false* for the key and value of the inner pair, since we do not utilize them as an actual key-value pair. As a matter of fact, we treat the inner pair as a plain tuple of values,.

Subsequently, we group all pairs by key (i.e., initial letter) and compute the following two intermediate results *per initial letter*:

(1) sum of word lengths that start with the respective letter, by adding up all false keys of the inner pair, and
(2) total number of words that start with the respective letter, by adding up all unary values (false values) of the inner pair.

Having computed these intermediate results for each initial letter, the average length can be computed as the fraction

$$average\ word\ length\ _{letter} = \frac{sum\ of\ word\ lengths\ _{letter}}{total\ number\ of\ words\ _{letter}}, \forall letter \in Latin\ Alphabet$$

As a final step, the results are sorted in descending order w.r.t. the average length and saved to a text file. By default, after successful execution of the program, the results can be found under the `output/task1` directory.

2.2.3 Evaluation. The experiments were executed using the dataset `SherlockHolmes.txt`. The results are listed below in a 5-column format and graphically depicted in Figure 1.

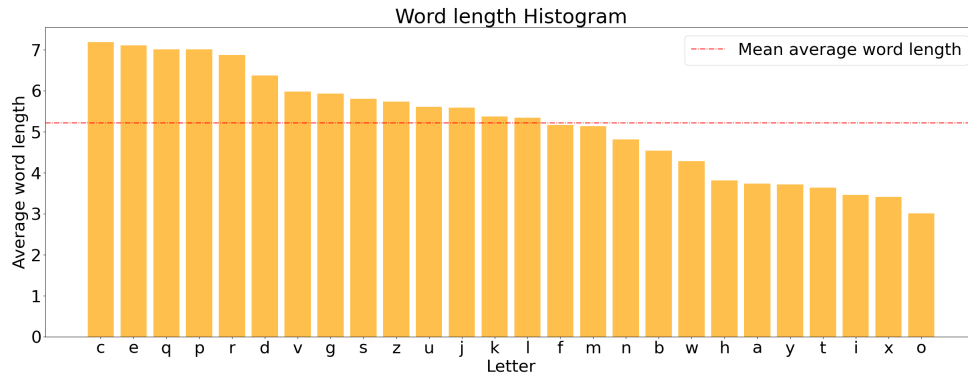|			
c:7.19	g: 5.93	f: 5.17	y: 3.72
e: 7.11	s: 5.8	m: 5.14	t: 3.64
q: 7.01	z: 5.74	n: 4.81	i: 3.46
p: 7.01	u: 5.61	b: 4.54	x: 3.41
r: 6.87	j: 5.59	w: 4.28	o: 3.01
d: 6.37	k: 5.37	h: 3.81	
v: 5.98	l: 5.34	a: 3.74	

Fig. 1. A histogram depicting the average word length (y-axis) for every letter of the Latin alphabet (x-axis). The mean average length is shown as a horizontal red dashed line.

3 Problem 2: Movie Analytics

The second problem focuses on producing analytic insights from an IMDB dataset.

3.1 Problem Statement

Implement a **MapReduce** program to perform analytics tasks on an IMDB dataset about movies. The utter goal of your analysis would be to extract useful insights from the available movie data that will assist the IMDB team provide better recommendations for movies, based on their genre and/or country. In particular, the dataset (`movies.csv`) contains the following fields:

- imdbID: unique identifier of the movie in the IMDB database
- title: the movie title
- year: the year the movie was first released
- duration: the duration of the movie
- genre(s): the genre or genres in which the movie is classified
- premier date: the date of the first showing of the movie
- score: the IMDB score of the movie
- country/-ies: the country or countries the movie was produced in

You are asked to implement Map-Reduce source code in Java programming language for the following analytics tasks:

- Calculate the total duration of all movies per country. Note that in case multiple countries are recorded for a movie, the respective duration should be counted for all of them separately.
- Calculate the total number of movies per year and genre, having IMDB score over 8. For movies that have more than one genre, the sum should be separate for each genre.

3.2 Proposed approach

3.2.1   Setting. Our implementation is run and tested in a Linux environment with 12 cores, using the Java programming language. We have used the Java Development Kit (JDK) version 11.0.11. The source code is developed in IntelliJ IDEA Community Edition 2021.1.1 and managed using Maven as the build tool.
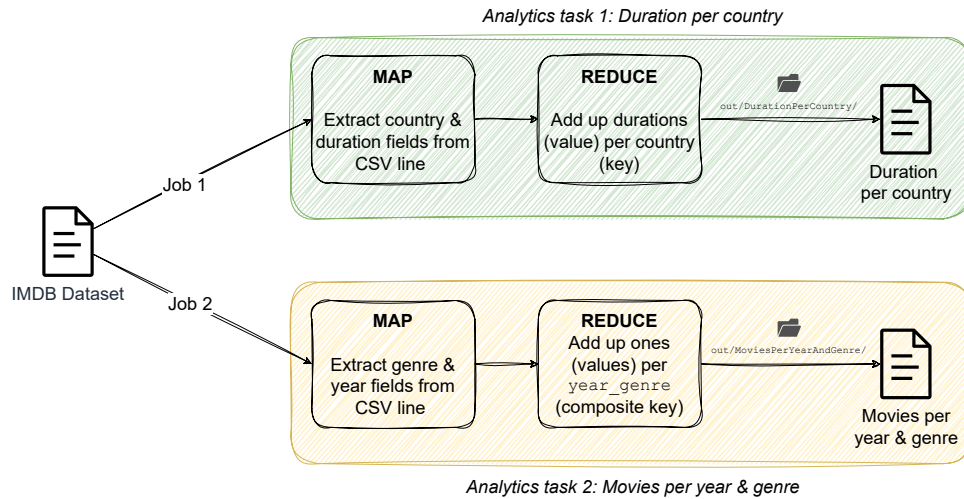
Fig. 2. The MapReduce solution architecture for the movie analytics problem. Two separate jobs are executed to solve the two tasks, namely duration per country (green) and movies per year & genre (yellow). The two tasks also create separate subdirectories for writing the results.

The project's dependencies, including Hadoop libraries, are defined in the `pom.xml` file located in the root of the repository. The project is compiled and executed directly from IntelliJ IDEA.

To run the project, open the `MovieAnalyticsMaster` class in IntelliJ IDEA, and execute the main method. You will need to specify the following command-line arguments:

- `<input_path>` specifies the directory containing the input text files, located at `map-reduce/movieAnalytics/input`.
- `<output_path>` specifies the directory where the MapReduce output will be written, which will be created in the `out` folder.

IntelliJ IDEA will handle the compilation and execution automatically when the main method is run. Make sure to configure the input and output paths as required for your specific run. Note that the command-line arguments must follow the specified order and format. If any of the arguments are missing or invalid, the program will terminate with an appropriate error message.

3.2.2 Implementation. The proposed approach leverages the MapReduce programming paradigm to compute the analytical insights for the two tasks. Figure 2 depicts the architecture of our solution as a diagram. We opt for executing two separate jobs; one for each task presented by the problem statement.

Task 1: Duration per country. One map-reduce cycle is enough to handle this task, as depicted in the top part of Figure 2 in green color. The map function parses the CSV file line by line and extracts the useful fields from each line. In this case, the useful fields are the country (or countries) the movie was produced and the movie duration, i.e., the fourth and ninth fields in the input line respectively. We employ more complex logic to handle cases where there are multiple countries in a single movie. In particular, we parse again the field and tokenize it into the separate countries, producing a key-value pair for each country-duration pair within a movie. The reduce function is, then, trivial; it just adds up the durations (value) per country (key) and outputs the

results. The interested reader can refer to the `CSVProcessor.java` (mapper), `AnayticsEngine.java` (reducer) and `MovieAnalyticsMaster.java` (driver) files for the source code implementation of our solution.

Task 2: Movies per year & genre w.r.t. score constraint. The task is similar to the first one, with the only difference lying in the fields extracted from the input CSV line. In this case, we are interested about the year, the genre (or genres) and score fields, i.e., the third, fifth and seventh fields in the input CSV line. The map function produces key-value pairs in the form (`composite_key, 1`), where the composite key consists of the year and the genre of the respective movie concatenated by an underscore, e.g., `2024_action`. We handle multiple genres per movie similarly with the multiple countries in task 1. We employ the same trivial reducer that adds up the values (ones) per (composite) key, as shown in the bottom part of Figure 2 in yellow color.

3.2.3 Evaluation. Our solution is tested using the provided IMDB dataset, namely `movies.csv`. We list the execution results in Appendix B.

4 Problem 3: DNA Sequence Patterns

The third problem focuses on mining unstructured data, namely DNA sequences, encoded in text format.

4.1 Problem Statement

A DNA sequence consists of four distinct symbols, namely A, G, C, and T. For instance, the following lines represent a part of the DNA sequence of the E. Coli bacterium [3]:

```
AGCTTTTCATTCTGACTGCAACGGGCAATATGTCTCTGTGTGGATTAAAAAAAGAGTGTCTGATAGCAGC
TTCTGAACTGGTTACCTGCCGTGAGTAAATTAAAATTTTATTGACTTAGGTCACTAAATACTTTAACCAA
TATAGGCATAGCGCACAGACAGATAAAAATTACAGAGTACACAACATCCATGAAACGCATTAGCACCACC
ATTACCACCACCATCACCATTACCACAGGTAACGGTGCGGGCTGACGCGTACAGGAAACACAGAAAAAAG
```

Implement a **MapReduce** program that computes the frequency (i.e., number of occurrences) of all subsequences of lengths 2, 3, and 4 that are present in the input DNA sequence. Note that the processing of each line should be independent of all other lines. Use the file `ecoli.txt` to test your solution.

4.2 Proposed approach

4.2.1 Setting. Our implementation is run and tested in a Linux environment with 12 cores, using the Java programming language. We have used the Java Development Kit (JDK) version 11.0.11. The source code is developed in IntelliJ IDEA Community Edition 2021.1.1 and managed using Maven as the build tool.

The project's dependencies, including Hadoop libraries, are defined in the `pom.xml` file located in the root of the repository. The project is compiled and executed directly from IntelliJ IDEA. To run the project, open the `DNASequenceAnalyticsMaster` class in IntelliJ IDEA, and execute the main method. You will need to specify the following command-line arguments:

- `<input_path>` specifies the directory containing the input text files, located at `map-reduce/dnaSequencePatterns/input`.
- `<output_path>` specifies the directory where the MapReduce output will be written, which will be created in the `out` folder.

IntelliJ IDEA will handle the compilation and execution automatically when the main method is run. Make sure to configure the input and output paths as required for your specific run. Note that the command-line arguments must follow the specified order and format. If any of the arguments are missing or invalid, the program will terminate with an appropriate error message.

---

[3]https://en.wikipedia.org/wiki/Pathogenic_Escherichia_coli

Algorithm 1 Populating DNA subsequences

1: ▷ *Input: a DNA sequence line (variable line)*   ◁
2: **for** *length* ∈ {2, 3, 4} **do**
3:      Create a sliding window *window* of length *length*.
4:      Place left end of *window* at the start of *line*.
5:      **while** right end of *window* has not exceeded *line* **do**
6:          *subsequence* ← contents of *window*
7:          Yield (*subsequence*, 1) key-value pair.
8:          Slide window to the right by 1.

4.2.2 Implementation. Algorithm 1 summarizes the most substantial part of our implementation, namely the logic that is implemented inside the mappers of our solution. At every execution of the map function, a DNA sequence line is given as input to the mapper. The latter is responsible for populating all possible subsequences of length 2, 3 and 4. We employ a sliding window that is nested inside a for-loop defining the subsequence length. The source code implementation of Algorithm 1 can be found in the `DNASequenceProcessor.java` file.

4.2.3 Evaluation. Our solution is tested using the provided DNA sequence, namely `ecoli.txt`. We list the execution results in Appendix C.

5 Problem 4: Probabilistic graph

We discuss here the fourth problem of the assignment. The main target of the assignment is to get acquainted with performing various MapReduce phases on a pipeline frame.

5.1 Problem Statement

We are given an input file in text format where each line contains a connection between two vertices of a network and a probability value. For each edge e, there is a probability value $p(e)$ which indicates the probability that the two vertices are connected by the edge. Obviously, the values of $p(e)$ range between 0 and 1. The values in each line are separated by a space. Consider the network shown in the previous figure. The edge connecting vertices 4 and 5 has a probability of 0.8, the edge connecting vertices 2 and 3 has a probability of 0.2, etc.
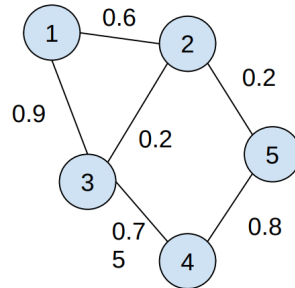
The file corresponding to this graph would be:

1 2 0.6, 1 3 0.9, 2 3 0.2, 3 4 0.75, 2 5 0.2, 4 5 0.8,

The edges are generally stored in random order in the file, so we cannot assume they have a specific arrangement. The following tasks are requested:

(1) Write a Java program that computes the average degree for all the vertices.
(2) The average degree is defined as the sum of the probabilities of the edges that fall on a vertex.
(3) For example, in the previous diagram, the average degree of vertex 3 is $0.9 + 0.2 + 0.75 = 1.85$.
(4) Before performing this calculation, you should ignore all edges with a probability less than a threshold T, which should be passed as a parameter to the main function.
(5) Modify the code from task 1 so that, at the end, only the vertices with an average degree greater than the average of the degrees of all the vertices are displayed in the output.

5.2 Proposed approach

5.2.1 Setting. Our implementation is run and tested in a Linux environment with 12 cores, using the Java programming language. We have used the Java Development Kit (JDK) version 11.0.11. The source code is developed in IntelliJ IDEA Community Edition 2021.1.1 and managed using Maven as the build tool.

The project's dependencies, including Hadoop libraries, are defined in the `pom.xml` file located in the root of the repository. The project is compiled and executed directly from IntelliJ IDEA.

To run the project, open the `GraphMaster` class in IntelliJ IDEA, and execute the main method. You will need to specify the following command-line arguments:

- `<input_path>` specifies the directory containing the input text files, located at `map-reduce/ probabilisticGraph /input`.
- `<output_path>` specifies the directory where the MapReduce output will be written, which will be created in the `out` folder.
- `<T>` is the minimum edge-degree threshold for vertices to be included in the results.

IntelliJ IDEA will handle the compilation and execution automatically when the main method is run. Make sure to configure the input and output paths as required for your specific run.

Note that the command-line arguments must follow the specified order and format. If any of the arguments are missing or invalid, the program will terminate with an appropriate error message.

5.2.2 Implementation. The problem at hand involves processing a probabilistic graph, where each edge connects two vertices with a probability value. The task is to compute the average degree for each vertex, considering only edges whose probability exceeds a given threshold T. The methodology proposed here utilizes a distributed MapReduce framework, implemented using Hadoop, to efficiently process and analyze large-scale graph data. The approach is divided into three main phases, each handled by separate MapReduce jobs, with intermediate results passed between the phases.

The `GraphMaster` class serves as the orchestrator of the entire MapReduce process. It manages the execution of the three phases, invoking the appropriate MapReduce jobs in sequence. The main steps executed by `GraphMaster` are:

(1) It retrieves the command-line arguments, including the input and output directories as well as the threshold value T, which determines which edges to consider in the graph.
(2) It initiates the first MapReduce job to compute the degree of each vertex in the graph. The input consists of edge data, and the output is a summation of edge probabilities for each vertex.
(3) After the first job completes, the `GraphMaster` starts the second MapReduce job to calculate the mean degree of the graph, using the results from the first job.
(4) Once the mean degree is computed, `GraphMaster` starts the third MapReduce job, which filters out the vertices with degrees lower than the mean degree.
(5) It cleans up intermediate directories after each phase and moves the final output to the desired location.

`GraphMaster` coordinates these steps by configuring and executing the MapReduce jobs, ensuring that each phase depends on the results of the previous one.

The entire process consists of three MapReduce jobs, executed sequentially:

(1) **Phase 1:** The first job computes the degree of each vertex by summing the probabilities of the edges connected to it. It produces intermediate results for each vertex.
(2) **Phase 2:** The second job calculates the mean degree by summing the degrees from the first phase and dividing by the total number of vertices.
(3) **Phase 3:** The third job filters out the vertices with a degree lower than the mean degree and produces the final filtered result.

After each job completes, intermediate data is written to disk and passed as input to the subsequent job. The final output consists of the vertices whose degree exceeds or equals the mean degree.

Phase 1: Calculating Vertex Degree The first MapReduce job is responsible for calculating the degree of each vertex in the graph, where the degree is defined as the sum of the probabilities of the edges connected to that vertex. The degree of each vertex is computed by the `GraphMapper` and `GraphReducer` classes.

The `GraphMapper` class reads each edge from the input data, which consists of lines containing two vertices and the associated probability value. It splits each line into two components (the two vertices), and for each vertex, it emits the corresponding probability value as the output. The mapper also checks if the probability value exceeds the threshold T, which is provided as a configuration parameter. If the probability is greater than or equal to T, it emits the vertex along with the corresponding probability.

The `GraphReducer` class receives the vertex and associated probability values emitted by the mapper. For each vertex, it sums up all the probabilities of the edges connected to that vertex, and emits the pair as output.

Phase 2: Calculating the Mean Degree The second phase of the approach calculates the mean degree of all vertices. This is done by summing the degree values from the previous phase and dividing by the total number of vertices.

The `MeanMapper` class receives the degree values from the output of the first MapReduce job. It emits two key-value pairs for each input line: a count of the number of vertices and the sum of the degree values. These are emitted with fixed keys (`"count"` and `"sum"`) so that they can be aggregated in the reducer.

The `MeanReducer` class processes the count and sum values emitted by the mapper. It calculates the total sum and count of vertices and then writes these values as output. The mean degree is calculated by dividing the sum by the count, and the result is stored for use in the next phase.

Phase 3: Filtering Vertices by Mean Degree In the third phase, the vertices whose degree is greater than or equal to the mean degree calculated in Phase 2 are selected. This phase uses the results from Phase 1 and Phase 2, applying the filtering criteria to retain only those vertices with a degree greater than or equal to the mean.

The `FilterMapper` class takes the degree values emitted by the first reducer and writes them as key-value pairs. Each key is a vertex, and the value is the degree of that vertex. The mapper does not perform any filtering; instead, it simply passes the values along to the reducer.

The `FilterReducer` class filters out the vertices whose degree is less than the mean degree. It retrieves the mean degree from the configuration, and for each vertex, it compares the degree value to the mean. If the degree is greater than or equal to the mean, it emits the vertex along with its degree. This final output contains only the vertices that satisfy the filtering criteria.

5.2.3 Evaluation. The experiments were executed using the dataset 'collins.txt' with a parameter setting of $T = 0.8$.

To provide a comprehensive overview, the results of the experiment, including the detailed values derived from the dataset, are presented in subsection C.1 in a 2-column format.

6 Conclusion

In this document we have presented our solutions and rationale for solving the second assignment of the M.Sc. course on *Technologies for Big Data Analysis*, offered by the *DWS M.Sc. Program*. For each one of the four problems of the assignment, we have presented their statement, as well as the solution approach we have adopted. All execution results with the provided datasets can be found in the appendices of our work.

A Problem 1 execution results

In the following 8-columned pages, the experimental results of section's 2 problem are presented, as the raw output of the MapReduce execution.

a10c: 18	a3a: 41	a6a: 28	a9c: 36	b2y: 459	b5m: 61	b8r: 33	c14e: 39
a10d: 94	a3d: 1360	a6c: 37	a9d: 165	b3a: 77	b5n: 861	b8s: 227	c14f: 10
a10e: 95	a3e: 1377	a6d: 1790	a9e: 136	b3d: 934	b5o: 15	b8t: 43	c1b: 34
a10g: 47	a3g: 841	a6e: 386	a9g: 101	b3e: 330	b5r: 238	b8y: 10	c1n: 872
a10l: 20	a3i: 11	a6g: 625	a9n: 202	b3f: 29	b5s: 741	b9a: 10	c1p: 117
a10n: 65	a3l: 112	a6h: 328	a9p: 16	b3g: 1080	b5t: 494	b9d: 79	c1t: 210
a10s: 139	a3m: 56	a6l: 125	a9r: 13	b3h: 103	b5u: 16	b9g: 15	c1y: 140
a10t: 59	a3n: 916	a6m: 239	a9s: 266	b3k: 342	b5y: 196	b9m: 34	c2a: 35
a10y: 36	a3r: 1604	a6n: 982	a9t: 115	b3l: 11	b6a: 121	b9n: 21	c2b: 72
a11e: 10	a3s: 213	a6r: 98	a9y: 119	b3m: 29	b6d: 308	b9o: 48	c2d: 333
a11g: 19	a3t: 1767	a6s: 469	b10d: 36	b3n: 1190	b6e: 83	b9s: 75	c2e: 2548
a11n: 22	a3w: 94	a6t: 440	b10e: 20	b3r: 60	b6g: 451	b9t: 13	c2f: 18
a11s: 112	a3y: 201	a6x: 10	b10g: 10	b3s: 1288	b6h: 15	b9y: 19	c2k: 36
a11t: 36	a4a: 112	a6y: 328	b10l: 18	b3t: 212	b6l: 55	c10d: 107	c2l: 246
a11y: 36	a4d: 785	a7a: 27	b10s: 25	b3w: 111	b6m: 41	c10e: 284	c2m: 88
a12c: 12	a4e: 615	a7c: 23	b10t: 10	b3y: 101	b6n: 53	c10g: 46	c2n: 92
a12e: 16	a4g: 214	a7d: 531	b10w: 10	b4a: 42	b6o: 105	c10h: 11	c2p: 105
a12n: 114	a4h: 21	a7e: 369	b10y: 21	b4d: 865	b6r: 55	c10l: 15	c2r: 31
a12s: 16	a4k: 151	a7g: 548	b11e: 10	b4e: 3041	b6s: 795	c10n: 673	c2s: 42
a12t: 17	a4l: 192	a7l: 62	b11v: 13	b4f: 52	b6t: 62	c10r: 10	c2t: 739
a12y: 38	a4m: 20	a7n: 555	b1d: 352	b4g: 84	b6v: 148	c10s: 173	c2w: 12
a13s: 33	a4n: 456	a7r: 235	b1e: 11	b4h: 123	b6y: 78	c10t: 45	c2y: 285
a14s: 18	a4p: 84	a7s: 622	b1g: 145	b4i: 20	b7a: 26	c10y: 124	c3a: 45
a1d: 38224	a4r: 523	a7t: 287	b1r: 22	b4l: 39	b7d: 95	c11d: 80	c3c: 10
a1e: 3767	a4s: 1197	a7v: 38	b1t: 5674	b4m: 41	b7e: 232	c11e: 35	c3d: 2428
a1k: 248	a4t: 811	a7y: 286	b1w: 35	b4n: 329	b7g: 271	c11g: 61	c3e: 804
a1l: 4077	a4w: 1078	a8c: 34	b1x: 66	b4r: 436	b7h: 18	c11l: 29	c3f: 320
a1m: 353	a4y: 204	a8d: 486	b1y: 223	b4s: 390	b7i: 161	c11n: 220	c3g: 24
a1o: 106	a5a: 431	a8e: 386	b2d: 268	b4t: 347	b7l: 175	c11s: 379	c3h: 168
a1r: 198	a5c: 52	a8g: 175	b2e: 883	b4u: 17	b7n: 274	c11y: 83	c3i: 18
a1t: 428	a5d: 1076	a8h: 58	b2f: 10	b4w: 15	b7r: 19	c12a: 14	c3k: 237
a1y: 1193	a5e: 685	a8l: 131	b2g: 96	b4y: 181	b7s: 163	c12c: 94	c3l: 318
a2a: 458	a5g: 71	a8m: 15	b2h: 538	b5a: 31	b7t: 150	c12e: 47	c3m: 77
a2d: 166	a5h: 12	a8n: 260	b2k: 969	b5c: 25	b7y: 19	c12l: 93	c3n: 161
a2e: 249	a5l: 116	a8p: 48	b2l: 288	b5d: 248	b8a: 17	c12n: 20	c3p: 33
a2h: 13	a5n: 83	a8r: 68	b2m: 16	b5e: 1005	b8d: 86	c12s: 84	c3r: 479
a2n: 15	a5r: 843	a8s: 429	b2n: 2688	b5g: 339	b8e: 58	c12y: 29	c3s: 1360
a2o: 773	a5s: 1203	a8t: 72	b2r: 102	b5h: 371	b8g: 28	c13d: 11	c3t: 1031
a2s: 483	a5t: 1011	a8y: 190	b2s: 119	b5i: 31	b8l: 20	c13e: 12	c3y: 130
a2t: 47	a5x: 17	a8z: 50	b2t: 458	b5k: 28	b8m: 12	c13s: 24	c4a: 44
a2y: 1588	a5y: 756	a9a: 36	b2w: 98	b5l: 16	b8n: 13	c13y: 18	c4c: 40

c4d: 871	c7l: 41	d12s: 10	d4i: 14	d7v: 26	e1t: 67	e5t: 209	e9y: 115
c4e: 1144	c7m: 42	d12t: 11	d4l: 146	d7y: 257	e2a: 12	e5v: 28	f10g: 14
c4g: 247	c7n: 616	d12y: 13	d4m: 27	d8a: 113	e2e: 304	e5y: 234	f10n: 11
c4h: 125	c7r: 485	d13n: 31	d4n: 177	d8c: 142	e2h: 409	e6a: 11	f10s: 33
c4l: 61	c7s: 836	d13s: 19	d4r: 617	d8d: 345	e2l: 53	e6c: 152	f11s: 10
c4m: 30	c7t: 198	d1c: 10	d4s: 281	d8e: 169	e2n: 941	e6d: 864	f1d: 18
c4n: 676	c7v: 10	d1d: 1874	d4t: 341	d8g: 194	e2r: 253	e6e: 572	f1e: 47
c4r: 706	c7y: 381	d1e: 335	d4y: 154	d8l: 29	e2s: 1113	e6g: 232	f1g: 340
c4s: 929	c8a: 49	d1g: 71	d5b: 13	d8n: 175	e2t: 149	e6l: 107	f1n: 26
c4t: 568	c8d: 460	d1m: 55	d5c: 21	d8r: 15	e2y: 129	e6m: 19	f1o: 17
c4x: 23	c8e: 395	d1n: 29	d5d: 989	d8s: 211	e3a: 13	e6n: 348	f1r: 7246
c4y: 363	c8g: 362	d1s: 11	d5e: 1011	d8t: 120	e3d: 43	e6r: 62	f1t: 128
c5a: 23	c8l: 129	d1y: 882	d5g: 439	d8v: 44	e3e: 56	e6s: 531	f1w: 453
c5c: 148	c8n: 689	d2a: 19	d5l: 18	d8y: 259	e3h: 117	e6t: 40	f1x: 44
c5d: 1329	c8r: 18	d2d: 284	d5n: 43	d9a: 13	e3k: 85	e6y: 174	f1y: 33
c5e: 561	c8s: 818	d2e: 602	d5o: 10	d9d: 205	e3l: 164	e7c: 42	f2d: 493
c5g: 328	c8t: 97	d2g: 27	d5p: 56	d9e: 106	e3n: 46	e7d: 544	f2e: 2351
c5h: 15	c8y: 256	d2h: 19	d5r: 107	d9g: 78	e3r: 210	e7e: 386	f2g: 31
c5k: 80	c9a: 21	d2k: 187	d5s: 665	d9h: 32	e3s: 98	e7g: 244	f2h: 16
c5l: 690	c9d: 116	d2l: 170	d5t: 220	d9l: 23	e3t: 327	e7h: 75	f2i: 27
c5m: 35	c9e: 272	d2m: 29	d5v: 366	d9m: 17	e3w: 87	e7l: 122	f2k: 35
c5n: 703	c9g: 141	d2n: 1183	d5y: 183	d9n: 352	e3y: 1240	e7n: 283	f2l: 859
c5o: 27	c9l: 92	d2p: 258	d6a: 46	d9s: 176	e4a: 13	e7r: 31	f2m: 6376
c5r: 658	c9n: 403	d2r: 935	d6c: 93	d9t: 121	e4d: 174	e7s: 330	f2p: 28
c5s: 834	c9p: 12	d2s: 900	d6d: 903	d9y: 63	e4e: 651	e7t: 153	f2r: 444
c5t: 612	c9r: 16	d2t: 713	d6e: 536	e10a: 11	e4g: 39	e7y: 634	f2s: 73
c5x: 23	c9s: 341	d2u: 19	d6g: 305	e10c: 16	e4h: 189	e8c: 57	f2t: 1443
c5y: 1192	c9t: 74	d2w: 203	d6h: 52	e10d: 49	e4i: 12	e8d: 104	f2w: 96
c6a: 125	c9y: 181	d2y: 161	d6k: 15	e10g: 79	e4m: 10	e8e: 256	f2x: 11
c6c: 46	d10d: 112	d3a: 11	d6l: 119	e10l: 13	e4n: 28	e8g: 553	f2y: 21
c6d: 582	d10e: 70	d3d: 114	d6m: 37	e10n: 38	e4r: 335	e8h: 23	f3a: 22
c6e: 638	d10g: 44	d3e: 296	d6n: 272	e10s: 88	e4s: 496	e8l: 67	f3d: 1113
c6g: 570	d10l: 21	d3g: 250	d6r: 296	e10t: 23	e4t: 710	e8m: 91	f3e: 402
c6h: 26	d10n: 162	d3h: 374	d6s: 475	e10y: 20	e4y: 213	e8n: 431	f3g: 32
c6l: 573	d10s: 168	d3k: 109	d6t: 183	e11n: 15	e5a: 18	e8s: 136	f3h: 404
c6n: 523	d10t: 32	d3l: 77	d6v: 284	e11s: 59	e5c: 69	e8t: 195	f3k: 140
c6o: 18	d10y: 37	d3m: 43	d6y: 138	e11t: 75	e5d: 1198	e8y: 610	f3l: 132
c6r: 285	d11d: 84	d3n: 209	d7a: 31	e11y: 91	e5e: 588	e9a: 49	f3n: 23
c6s: 2081	d11e: 52	d3s: 415	d7d: 793	e12g: 10	e5g: 279	e9d: 331	f3r: 353
c6t: 341	d11g: 13	d3t: 305	d7e: 326	e12s: 24	e5h: 213	e9e: 20	f3s: 867
c6x: 16	d11n: 63	d3y: 245	d7g: 169	e12y: 14	e5l: 29	e9g: 29	f3t: 1749
c6y: 394	d11s: 29	d4a: 39	d7m: 14	e1a: 11	e5m: 11	e9l: 62	f3y: 346
c7a: 39	d11t: 13	d4d: 313	d7n: 202	e1c: 18	e5n: 255	e9m: 20	f4a: 61
c7d: 1289	d12d: 17	d4e: 549	d7r: 48	e1d: 456	e5o: 10	e9n: 195	f4d: 958
c7e: 440	d12g: 16	d4g: 566	d7s: 855	e1e: 137	e5r: 569	e9s: 134	f4e: 500
c7g: 137	d12n: 27	d4h: 10	d7t: 470	e1r: 45	e5s: 575	e9t: 95	f4g: 183

f4h: 1159	f7y: 59	g3e: 276	g7a: 19	h2l: 277	h6c: 47	i10y: 145	i4t: 119
f4l: 60	f8d: 228	g3f: 47	g7d: 33	h2m: 38	h6d: 313	i11e: 100	i4y: 127
f4n: 122	f8e: 83	g3g: 369	g7e: 65	h2n: 22	h6e: 182	i11l: 51	i5a: 32
f4o: 13	f8g: 95	g3h: 15	g7g: 188	h2o: 53	h6g: 47	i11n: 35	i5d: 420
f4r: 892	f8h: 29	g3i: 10	g7n: 179	h2p: 259	h6h: 10	i11r: 20	i5e: 352
f4s: 520	f8n: 128	g3m: 59	g7r: 45	h2r: 557	h6l: 66	i11s: 32	i5g: 18
f4t: 319	f8p: 42	g3n: 594	g7s: 160	h2s: 66	h6n: 150	i11t: 33	i5h: 14
f4w: 344	f8r: 20	g3p: 168	g7t: 37	h2t: 229	h6o: 22	i11y: 222	i5l: 35
f4y: 435	f8s: 124	g3s: 549	g7y: 190	h2y: 50	h6s: 254	i12e: 26	i5n: 37
f5a: 76	f8t: 63	g3t: 935	g8a: 20	h3d: 709	h6t: 33	i12n: 20	i5s: 296
f5d: 590	f8v: 12	g3y: 136	g8d: 16	h3e: 1028	h6y: 89	i12s: 16	i5t: 134
f5e: 324	f8y: 242	g4a: 13	g8g: 52	h3h: 23	h7a: 19	i12y: 13	i5y: 44
f5g: 756	f9d: 19	g4d: 281	g8n: 37	h3i: 10	h7c: 10	i13y: 17	i6a: 25
f5h: 48	f9e: 46	g4e: 336	g8r: 12	h3l: 22	h7d: 139	i14e: 39	i6d: 845
f5l: 282	f9g: 73	g4g: 256	g8s: 120	h3n: 184	h7e: 136	i1e: 78	i6e: 317
f5m: 220	f9l: 21	g4h: 223	g8t: 643	h3o: 10	h7g: 70	i1i: 91	i6g: 88
f5n: 410	f9n: 42	g4l: 22	g8y: 40	h3r: 193	h7k: 10	i1k: 12	i6l: 221
f5r: 333	f9s: 142	g4n: 305	g9d: 60	h3s: 1072	h7n: 61	i1l: 278	i6m: 41
f5s: 1066	f9t: 15	g4p: 63	g9g: 22	h3t: 311	h7s: 256	i1n: 24	i6n: 96
f5t: 43	f9y: 59	g4r: 47	g9l: 21	h3y: 509	h7t: 18	i1s: 2045	i6r: 90
f5y: 265	g10d: 10	g4s: 717	g9m: 126	h4a: 30	h7y: 121	i2a: 188	i6s: 247
f6d: 604	g10l: 16	g4t: 10	g9n: 89	h4c: 33	h8a: 55	i2e: 25	i6t: 414
f6e: 190	g10s: 74	g4y: 229	g9r: 20	h4d: 321	h8c: 17	i2h: 11	i6y: 305
f6g: 218	g11y: 10	g5a: 83	g9s: 71	h4e: 212	h8d: 35	i2n: 128	i7a: 14
f6h: 39	g12g: 10	g5d: 366	h10a: 17	h4g: 718	h8e: 22	i2o: 2120	i7d: 418
f6k: 11	g16g: 13	g5e: 126	h10c: 12	h4h: 170	h8g: 26	i2s: 12	i7e: 538
f6l: 56	g1d: 305	g5g: 314	h10f: 55	h4i: 14	h8h: 21	i2t: 50	i7g: 250
f6m: 21	g1m: 13	g5h: 19	h10g: 12	h4n: 214	h8l: 76	i3a: 23	i7h: 35
f6n: 226	g1n: 54	g5l: 848	h10s: 90	h4r: 443	h8n: 44	i3c: 25	i7l: 24
f6r: 152	g1p: 44	g5m: 25	h11d: 13	h4s: 1051	h8s: 207	i3e: 118	i7n: 557
f6s: 473	g1s: 11	g5n: 29	h11s: 20	h4t: 96	h8y: 38	i3h: 20	i7r: 89
f6t: 118	g1t: 752	g5p: 15	h1d: 7393	h4w: 55	h9a: 19	i3l: 32	i7s: 381
f6y: 223	g1y: 33	g5r: 167	h1e: 11	h4y: 274	h9c: 13	i3n: 39	i7t: 371
f7a: 17	g2d: 855	g5s: 328	h1m: 5193	h5a: 16	h9d: 17	i3r: 65	i7w: 34
f7c: 12	g2e: 1349	g5t: 31	h1p: 37	h5d: 544	h9e: 143	i3s: 91	i7y: 124
f7d: 147	g2f: 13	g5y: 216	h1r: 5252	h5e: 44	h9g: 21	i3t: 20	i8d: 309
f7e: 131	g2l: 185	g6c: 29	h1s: 11727	h5f: 1487	h9n: 16	i3x: 17	i8e: 645
f7g: 266	g2m: 20	g6d: 221	h1t: 268	h5g: 535	h9s: 32	i3y: 73	i8g: 134
f7h: 26	g2n: 91	g6e: 185	h1w: 1296	h5h: 13	h9y: 13	i4a: 11	i8l: 234
f7k: 31	g2p: 14	g6g: 232	h1y: 47	h5l: 48	i10d: 19	i4c: 10	i8n: 306
f7l: 51	g2s: 269	g6l: 37	h2d: 2139	h5m: 25	i10e: 359	i4d: 291	i8s: 253
f7n: 454	g2t: 44	g6n: 73	h2e: 4797	h5n: 48	i10g: 33	i4e: 216	i8t: 99
f7o: 16	g2w: 255	g6r: 139	h2f: 251	h5r: 466	i10l: 55	i4f: 264	i8y: 163
f7r: 24	g2y: 110	g6s: 306	h2g: 59	h5s: 490	i10n: 505	i4m: 34	i9d: 212
f7s: 309	g3a: 40	g6t: 83	h2h: 286	h5t: 118	i10s: 141	i4n: 64	i9e: 172
f7t: 53	g3d: 278	g6y: 37	h2k: 11	h5y: 683	i10t: 52	i4s: 110	i9g: 127

i9l: 15	j7n: 108	k9a: 25	l4l: 31	l9n: 56	m4d: 390	m7e: 125	n2e: 683
i9m: 10	j7s: 24	k9d: 10	l4n: 315	l9s: 40	m4e: 621	m7g: 46	n2k: 193
i9n: 300	j7y: 17	k9n: 15	l4r: 930	l9y: 10	m4f: 223	m7h: 10	n2l: 58
i9r: 23	j8n: 23	l10a: 19	l4s: 314	m10d: 10	m4g: 365	m7i: 28	n2r: 286
i9s: 229	j8s: 33	l10d: 13	l4t: 52	m10e: 155	m4l: 117	m7l: 28	n2s: 237
i9t: 191	k1y: 23	l10g: 11	l4x: 25	m10h: 22	m4m: 62	m7m: 24	n2t: 281
i9y: 354	k2d: 194	l10n: 22	l4y: 241	m10n: 24	m4n: 227	m7n: 49	n2y: 35
j10n: 27	k2e: 135	l10s: 129	l5c: 10	m10r: 32	m4o: 72	m7o: 30	n3d: 75
j11e: 10	k2g: 211	l11n: 11	l5d: 391	m10s: 88	m4r: 1005	m7r: 36	n3e: 457
j11n: 15	k2l: 64	l11s: 19	l5e: 80	m10y: 31	m4s: 452	m7s: 505	n3h: 222
j1w: 33	k2n: 35	l1d: 274	l5g: 952	m11g: 38	m4t: 569	m7t: 124	n3i: 10
j1y: 103	k2p: 151	l1e: 109	l5l: 90	m11l: 11	m4u: 10	m7y: 92	n3l: 79
j2e: 80	k2s: 59	l1g: 157	l5n: 92	m11n: 29	m4v: 12	m8c: 11	n3o: 22
j2k: 10	k2t: 279	l1p: 63	l5r: 47	m11s: 160	m4w: 803	m8d: 82	n3r: 590
j2n: 208	k2v: 17	l1s: 20	l5s: 505	m12s: 65	m4x: 10	m8e: 72	n3s: 189
j2p: 13	k2w: 1529	l1t: 573	l5t: 46	m13s: 10	m4y: 443	m8g: 27	n3t: 335
j2s: 45	k3e: 45	l1w: 486	l5v: 10	m1b: 26	m5a: 163	m8l: 50	n3y: 58
j2t: 773	k3k: 12	l1y: 289	l5y: 211	m1d: 61	m5c: 23	m8m: 13	n4a: 20
j2y: 62	k3l: 48	l2b: 213	l6d: 221	m1n: 2654	m5d: 388	m8n: 58	n4d: 138
j3e: 106	k3n: 379	l2d: 791	l6e: 195	m1p: 40	m5e: 302	m8r: 12	n4e: 345
j3n: 22	k3s: 255	l2e: 3014	l6g: 201	m1s: 58	m5g: 624	m8s: 199	n4g: 13
j3s: 137	k3y: 18	l2g: 942	l6n: 79	m1t: 547	m5h: 27	m8t: 63	n4i: 10
j3t: 329	k4a: 26	l2h: 10	l6r: 97	m1y: 2535	m5l: 178	m8y: 69	n4l: 107
j3y: 14	k4d: 279	l2k: 658	l6s: 142	m2d: 514	m5m: 40	m9a: 133	n4n: 210
j4d: 170	k4r: 22	l2n: 42	l6t: 42	m2e: 3829	m5n: 265	m9d: 16	n4r: 379
j4e: 10	k4s: 52	l2p: 53	l6y: 75	m2h: 671	m5r: 38	m9e: 51	n4s: 243
j4g: 19	k4y: 118	l2s: 1182	l7a: 102	m2k: 98	m5s: 1188	m9g: 26	n4t: 10
j4h: 47	k5a: 20	l2t: 1734	l7c: 23	m2l: 32	m5t: 11	m9i: 81	n4w: 77
j4l: 53	k5d: 49	l2y: 164	l7d: 105	m2n: 233	m5y: 221	m9n: 10	n4y: 268
j4s: 239	k5e: 29	l3a: 17	l7e: 45	m2s: 396	m6a: 24	m9r: 10	n5a: 1119
j4t: 39	k5g: 213	l3c: 15	l7g: 121	m2t: 2077	m6c: 41	m9s: 90	n5d: 203
j4y: 49	k5h: 15	l3d: 371	l7n: 29	m2y: 1246	m6d: 222	m9t: 20	n5e: 76
j5e: 83	k5m: 24	l3e: 965	l7r: 46	m3a: 308	m6e: 505	m9y: 26	n5g: 662
j5g: 71	k5n: 117	l3g: 114	l7s: 137	m3c: 59	m6g: 123	n10d: 18	n5k: 12
j5n: 87	k5s: 39	l3h: 291	l7y: 53	m3d: 312	m6i: 56	n10g: 18	n5l: 164
j5r: 20	k5v: 448	l3l: 303	l8d: 24	m3e: 66	m6k: 29	n10s: 83	n5n: 10
j5s: 26	k6a: 21	l3n: 102	l8e: 52	m3h: 389	m6l: 137	n11d: 18	n5r: 179
j5y: 127	k6f: 11	l3r: 934	l8g: 13	m3l: 109	m6m: 17	n11n: 33	n5s: 364
j6e: 15	k6g: 26	l3s: 739	l8n: 42	m3m: 27	m6n: 158	n11s: 12	n5t: 73
j6g: 12	k6s: 131	l3t: 480	l8p: 56	m3n: 51	m6r: 283	n13d: 12	n5y: 71
j6l: 26	k6v: 70	l3y: 49	l8s: 135	m3o: 12	m6s: 535	n15s: 13	n6a: 25
j6n: 52	k6y: 70	l4a: 41	l8t: 27	m3r: 150	m6t: 448	n1p: 10	n6d: 33
j6s: 46	k7e: 71	l4d: 1083	l8y: 15	m3s: 701	m6y: 608	n1r: 275	n6e: 134
j6t: 34	k7i: 17	l4e: 1215	l9d: 16	m3t: 832	m7a: 25	n1t: 6590	n6g: 64
j6y: 72	k7s: 32	l4g: 269	l9e: 112	m3y: 567	m7c: 18	n1w: 2862	n6l: 288
j7d: 21	k7t: 22	l4h: 66	l9g: 15	m4a: 46	m7d: 138	n2d: 156	n6n: 619

n6p: 11	o1r: 1064	o6e: 194	p11e: 19	p4c: 307	p7g: 257	q5y: 276	r2n: 90
n6r: 84	o1t: 1956	o6g: 69	p11g: 10	p4d: 1413	p7h: 43	q6d: 17	r2o: 22
n6s: 932	o2d: 13	o6h: 12	p11k: 11	p4e: 4994	p7l: 405	q6n: 339	r2p: 11
n6y: 47	o2e: 567	o6k: 37	p11l: 20	p4g: 85	p7m: 20	q6s: 77	r2r: 49
n7a: 35	o2h: 27	o6l: 189	p11n: 38	p4h: 106	p7n: 169	q6y: 38	r2s: 142
n7d: 68	o2l: 24	o6m: 34	p11r: 22	p4l: 75	p7o: 13	q7d: 28	r2t: 262
n7e: 56	o2n: 302	o6n: 59	p11s: 62	p4n: 277	p7r: 73	q7g: 18	r2y: 39
n7g: 16	o2o: 159	o6r: 29	p11y: 20	p4o: 11	p7s: 633	q7s: 218	r3d: 675
n7i: 66	o2r: 1263	o6s: 467	p12s: 14	p4r: 213	p7t: 494	q8d: 23	r3e: 195
n7l: 17	o2s: 213	o6t: 117	p13s: 11	p4s: 813	p7y: 262	q8g: 13	r3h: 130
n7n: 23	o2y: 1890	o6w: 18	p1a: 11	p4t: 205	p8a: 38	q8s: 20	r3l: 142
n7r: 23	o3a: 16	o6y: 100	p1d: 19	p4w: 26	p8c: 17	q9g: 27	r3m: 20
n7s: 168	o3d: 19	o7a: 13	p1n: 31	p4y: 342	p8d: 138	r10d: 15	r3n: 75
n7t: 53	o3e: 27	o7d: 111	p1r: 88	p5a: 93	p8e: 184	r10e: 26	r3r: 210
n7y: 486	o3g: 44	o7e: 220	p1s: 170	p5c: 94	p8g: 464	r10g: 29	r3s: 525
n8a: 11	o3m: 23	o7g: 218	p1t: 465	p5d: 592	p8h: 54	r10n: 115	r3t: 714
n8c: 16	o3n: 465	o7h: 11	p1y: 123	p5e: 843	p8l: 128	r10s: 72	r3w: 12
n8d: 20	o3r: 2249	o7n: 252	p2a: 67	p5g: 540	p8m: 121	r10y: 59	r3y: 426
n8e: 41	o3s: 20	o7p: 28	p2d: 122	p5k: 18	p8n: 521	r11d: 11	r4a: 185
n8g: 23	o3t: 173	o7s: 367	p2e: 410	p5l: 134	p8r: 124	r11e: 17	r4d: 791
n8h: 21	o3y: 12	o7t: 11	p2h: 115	p5m: 147	p8s: 413	r11n: 13	r4e: 290
n8k: 18	o4a: 76	o7w: 19	p2k: 99	p5n: 223	p8t: 142	r11s: 25	r4f: 65
n8n: 29	o4d: 243	o7y: 56	p2l: 60	p5r: 246	p8y: 317	r11y: 71	r4g: 174
n8s: 57	o4e: 181	o8d: 65	p2m: 39	p5s: 1374	p9c: 22	r12e: 39	r4k: 46
n8y: 30	o4g: 17	o8e: 91	p2n: 463	p5t: 1435	p9d: 50	r12g: 10	r4l: 115
n9g: 24	o4k: 100	o8g: 26	p2p: 11	p5x: 39	p9e: 166	r12n: 87	r4m: 135
n9m: 22	o4n: 171	o8h: 28	p2r: 186	p5y: 237	p9g: 88	r12s: 24	r4n: 775
n9n: 11	o4r: 13	o8l: 26	p2s: 205	p6a: 149	p9n: 117	r12y: 39	r4r: 415
n9s: 35	o4s: 913	o8n: 177	p2t: 1040	p6c: 136	p9r: 25	r13s: 104	r4s: 464
n9t: 16	o4t: 159	o8s: 177	p2y: 256	p6d: 819	p9s: 289	r1b: 10	r4t: 710
n9y: 62	o4y: 47	o8y: 31	p3a: 255	p6e: 1405	p9t: 24	r1d: 252	r4v: 715
o10a: 14	o4z: 22	o9d: 36	p3c: 17	p6g: 312	p9y: 240	r1e: 18	r4w: 43
o10d: 30	o5a: 31	o9g: 54	p3d: 98	p6h: 16	q11n: 10	r1g: 12	r4y: 402
o10g: 17	o5c: 14	o9n: 51	p3e: 1330	p6i: 16	q11r: 16	r1m: 24	r5a: 32
o10n: 94	o5d: 336	o9s: 94	p3f: 30	p6k: 12	q12s: 26	r1n: 458	r5d: 1139
o10s: 23	o5e: 273	o9y: 79	p3h: 143	p6l: 356	q1e: 16	r1p: 15	r5e: 542
o10y: 91	o5f: 26	p10a: 159	p3k: 10	p6m: 66	q2t: 11	r1t: 11	r5g: 496
o11s: 108	o5g: 151	p10d: 37	p3l: 69	p6n: 539	q3e: 500	r1w: 79	r5h: 25
o1d: 1140	o5k: 10	p10e: 46	p3n: 102	p6r: 160	q3k: 75	r2d: 472	r5i: 15
o1e: 3233	o5n: 232	p10g: 14	p3o: 26	p6s: 2343	q3n: 18	r2e: 1174	r5l: 179
o1f: 625	o5r: 491	p10l: 106	p3p: 31	p6t: 221	q3r: 13	r2f: 30	r5n: 502
o1g: 15	o5s: 236	p10n: 115	p3r: 766	p6y: 390	q3t: 145	r2g: 78	r5r: 111
o1h: 13	o5t: 11	p10r: 13	p3s: 870	p7a: 56	q4s: 12	r2h: 129	r5s: 1095
o1k: 27	o5y: 47	p10s: 131	p3t: 299	p7c: 35	q4y: 15	r2k: 124	r5t: 329
o1l: 33	o6a: 30	p10t: 18	p3y: 332	p7d: 714	q5l: 31	r2l: 129	r5y: 473
o1n: 777	o6d: 564	p10y: 223	p4a: 44	p7e: 216	q5r: 62	r2m: 905	r6c: 69

r6d: 2212	s10s: 239	s2r: 148	s5l: 555	s8l: 69	t2e: 2747	t5g: 579	t8t: 128
r6e: 325	s10t: 10	s2s: 446	s5m: 74	s8m: 50	t2k: 903	t5h: 891	t8y: 60
r6g: 237	s10y: 148	s2t: 1025	s5n: 390	s8n: 196	t2l: 849	t5l: 104	t9d: 112
r6h: 68	s11d: 49	s2w: 355	s5o: 14	s8o: 32	t2m: 2353	t5n: 101	t9e: 135
r6l: 24	s11e: 38	s2y: 85	s5r: 627	s8r: 44	t2n: 3322	t5o: 49	t9g: 36
r6n: 327	s11g: 17	s3a: 449	s5s: 1252	s8s: 469	t2p: 45	t5r: 119	t9h: 33
r6r: 184	s11l: 15	s3d: 886	s5t: 434	s8t: 204	t2r: 83	t5s: 882	t9l: 27
r6s: 560	s11n: 29	s3e: 2969	s5v: 34	s8y: 114	t2s: 4312	t5t: 950	t9n: 68
r6t: 254	s11r: 10	s3f: 189	s5w: 29	s9a: 24	t2t: 12263	t5y: 163	t9r: 11
r6y: 147	s11s: 67	s3g: 42	s5y: 655	s9c: 28	t2u: 40	t6a: 23	t9s: 324
r7c: 15	s11t: 43	s3h: 371	s6a: 29	s9d: 98	t2y: 3875	t6c: 32	t9t: 32
r7d: 317	s11y: 51	s3k: 522	s6c: 43	s9e: 98	t3a: 66	t6d: 414	t9y: 34
r7e: 186	s12e: 17	s3l: 2431	s6d: 882	s9g: 154	t3b: 50	t6e: 370	u10d: 29
r7g: 642	s12n: 11	s3m: 131	s6e: 550	s9h: 43	t3c: 30	t6g: 350	u10e: 43
r7m: 32	s12s: 29	s3n: 545	s6g: 1342	s9l: 141	t3d: 552	t6h: 15	u10s: 17
r7n: 141	s12t: 55	s3o: 17	s6h: 211	s9m: 17	t3e: 6460	t6l: 76	u10y: 82
r7r: 21	s12y: 47	s3p: 350	s6k: 203	s9n: 206	t3f: 10	t6n: 157	u11d: 21
r7s: 475	s13n: 10	s3r: 145	s6l: 219	s9r: 21	t3g: 303	t6o: 31	u11e: 32
r7t: 164	s13r: 10	s3s: 1274	s6m: 19	s9s: 151	t3h: 365	t6r: 292	u11g: 85
r7y: 15	s13y: 10	s3t: 1035	s6n: 509	s9t: 51	t3k: 820	t6s: 365	u11s: 10
r8d: 398	s14n: 14	s3w: 24	s6r: 325	s9w: 12	t3l: 88	t6t: 34	u11y: 43
r8e: 234	s14s: 10	s3y: 451	s6s: 1622	s9y: 108	t3n: 516	t6w: 120	u12e: 21
r8g: 175	s1a: 74	s4a: 44	s6t: 287	t10d: 29	t3p: 12	t6y: 235	u12y: 16
r8l: 75	s1b: 18	s4b: 12	s6v: 15	t10e: 58	t3r: 3002	t7c: 48	u13y: 10
r8m: 39	s1c: 72	s4c: 51	s6x: 30	t10g: 18	t3s: 1525	t7d: 151	u14l: 15
r8n: 640	s1d: 87	s4d: 3507	s6y: 1160	t10h: 23	t3t: 146	t7e: 291	u1e: 313
r8r: 11	s1e: 4950	s4e: 1353	s7a: 35	t10n: 10	t3w: 142	t7g: 139	u1m: 11
r8s: 168	s1m: 55	s4g: 824	s7c: 50	t10s: 121	t3y: 165	t7h: 45	u2a: 12
r8t: 29	s1n: 409	s4h: 368	s7d: 806	t10y: 42	t4a: 40	t7k: 31	u2d: 271
r8y: 99	s1r: 176	s4k: 162	s7e: 371	t11d: 10	t4d: 1075	t7l: 16	u2h: 18
r9d: 110	s1t: 801	s4l: 238	s7g: 1178	t11e: 14	t4e: 817	t7n: 39	u2n: 1106
r9e: 78	s1w: 600	s4m: 373	s7h: 14	t11n: 16	t4f: 115	t7o: 11	u2o: 13
r9g: 87	s1x: 160	s4n: 423	s7i: 67	t11s: 23	t4g: 502	t7r: 13	u3d: 52
r9h: 14	s1y: 858	s4r: 559	s7k: 21	t11y: 18	t4h: 658	t7s: 250	u3e: 192
r9l: 13	s2a: 109	s4s: 2097	s7l: 98	t12n: 32	t4l: 94	t7t: 406	u3g: 47
r9n: 151	s2b: 23	s4t: 927	s7m: 13	t12s: 22	t4n: 384	t7x: 17	u3l: 497
r9s: 259	s2d: 3697	s4w: 135	s7n: 429	t13n: 11	t4r: 481	t7y: 166	u3n: 265
r9t: 28	s2e: 3848	s4y: 689	s7o: 13	t1a: 104	t4s: 1255	t8a: 13	u3r: 1283
r9y: 55	s2f: 26	s5a: 153	s7r: 137	t1e: 79438	t4t: 198	t8d: 74	u3s: 17
s10d: 79	s2g: 123	s5b: 12	s7s: 1044	t1n: 223	t4x: 17	t8e: 100	u3t: 30
s10e: 107	s2h: 1441	s5c: 26	s7t: 194	t1o: 1608	t4y: 417	t8g: 49	u3y: 17
s10g: 40	s2k: 207	s5d: 1327	s7y: 486	t1p: 45	t5a: 30	t8h: 15	u4d: 608
s10h: 15	s2l: 309	s5e: 1364	s8c: 158	t1t: 20	t5c: 48	t8l: 32	u4e: 216
s10i: 17	s2m: 135	s5g: 1121	s8d: 321	t1x: 75	t5d: 340	t8n: 41	u4g: 15
s10n: 81	s2n: 1527	s5h: 163	s8e: 125	t1y: 137	t5e: 126	t8r: 53	u4l: 87
s10r: 15	s2p: 361	s5i: 13	s8g: 230	t2d: 590	t5f: 32	t8s: 521	u4n: 32

u4r: 19	u8g: 69	v3e: 630	v6g: 35	w1t: 64	w4h: 89	w7e: 35	y2k: 193
u4s: 336	u8l: 22	v3l: 45	v6h: 31	w1x: 21	w4m: 25	w7g: 81	y2r: 1539
u4t: 45	u8n: 74	v3r: 25	v6l: 18	w1y: 1522	w4n: 580	w7i: 29	y3d: 33
u4y: 26	u8r: 12	v3s: 296	v6n: 37	w2d: 469	w4r: 152	w7k: 20	y3e: 119
u5a: 10	u8s: 23	v3t: 134	v6r: 50	w2e: 4923	w4s: 495	w7l: 33	y3g: 622
u5d: 96	u8t: 56	v3x: 31	v6s: 242	w2f: 56	w4t: 114	w7n: 96	y3h: 65
u5e: 19	u8y: 71	v4a: 77	v6t: 25	w2g: 47	w4w: 187	w7r: 37	y3l: 76
u5g: 21	u9d: 54	v4d: 93	v6y: 104	w2h: 9956	w4y: 187	w7s: 72	y3s: 656
u5l: 37	u9e: 78	v4e: 97	v7a: 17	w2k: 625	w5d: 411	w7t: 28	y3v: 11
u5m: 119	u9g: 42	v4i: 214	v7e: 86	w2l: 2928	w5e: 173	w7y: 21	y4d: 11
u5n: 97	u9s: 41	v4l: 144	v7g: 20	w2m: 556	w5g: 745	w8d: 28	y4s: 21
u5o: 42	u9t: 15	v4m: 17	v7i: 12	w2n: 2930	w5l: 10	w8e: 13	y4w: 55
u5s: 93	u9y: 111	v4n: 29	v7l: 18	w2p: 68	w5m: 60	w8g: 28	y5d: 16
u5y: 591	v10n: 24	v4r: 17	v7n: 24	w2r: 28	w5n: 260	w8l: 18	y5g: 11
u6d: 65	v11h: 11	v4s: 372	v7r: 12	w2s: 160	w5r: 503	w8n: 180	y5r: 40
u6e: 44	v11t: 20	v4t: 33	v7s: 164	w2t: 4856	w5s: 213	w8r: 21	y6f: 160
u6g: 38	v1i: 32	v4y: 118	v7y: 46	w2y: 16	w5t: 1070	w8s: 62	y6g: 20
u6l: 16	v1l: 80	v5a: 37	v8d: 18	w3d: 2652	w5w: 34	w9d: 34	y6l: 15
u6n: 50	v1n: 27	v5d: 31	v8e: 18	w3e: 3284	w5y: 80	w9g: 21	y6n: 10
u6s: 67	v2a: 72	v5e: 312	v8n: 14	w3f: 11	w6d: 187	w9s: 45	y6t: 11
u6y: 68	v2d: 20	v5g: 42	v8r: 16	w3g: 116	w6e: 13	x1i: 74	y7l: 11
u7d: 228	v2e: 171	v5k: 10	v8s: 72	w3h: 4911	w6g: 108	x1v: 34	y7y: 56
u7e: 42	v2i: 39	v5n: 31	v8y: 46	w3l: 29	w6l: 16	x1x: 19	y8s: 20
u7g: 41	v2l: 21	v5o: 10	v9n: 12	w3n: 664	w6n: 44	x2i: 56	z2e: 23
u7l: 77	v2n: 136	v5r: 72	v9y: 12	w3r: 194	w6o: 19	x2y: 30	z2l: 24
u7n: 40	v2s: 60	v5s: 497	w10d: 11	w3s: 1155	w6r: 176	x3i: 33	z3m: 12
u7s: 56	v2t: 65	v5t: 68	w1b: 20	w3t: 165	w6s: 160	x3s: 53	z4r: 12
u7t: 10	v2w: 173	v5y: 265	w1n: 74	w3y: 84	w6t: 10	y1s: 675	z5s: 10
u7y: 56	v2y: 1394	v6a: 153	w1o: 2992	w4d: 743	w6w: 41	y1t: 476	z5v: 37
u8d: 775	v3a: 93	v6d: 92	w1r: 844	w4e: 45	w6y: 17	y1u: 5361	
u8e: 20	v3d: 61	v6e: 152	w1s: 11367	w4g: 66	w7d: 139	y2d: 94	

B Problem 2 execution results

We list here the execution results of our solution for the second problem (movie analytics) tested on the `movies.csv` dataset. Task 1 (duration per country) produces the following results:

Afghanistan: 815	Austria: 22925	Bolivia: 797	Cameroon: 615
Albania: 646	Bahamas: 560	Bosnia and Herzegovina: 2207	Canada: 210826
Algeria: 2115	Bahrain: 87		Chad: 493
American Samoa: 247	Bangladesh: 429	Botswana: 295	Chile: 6178
Angola: 357	Barbados: 92	Brazil: 27333	China: 42089
Argentina: 28813	Belarus: 427	Bulgaria: 5430	Colombia: 3810
Armenia: 264	Belgium: 54192	Burkina Faso: 903	Congo: 88
Aruba: 739	Bermuda: 95	Burma: 95	Costa Rica: 431
Australia: 66068	Bhutan: 291	Cambodia: 894	Croatia: 4340

Cuba: 2899
Cyprus: 948
Czech Republic: 15317
Czechoslovakia: 8434
Cóte d'Ivoire: 180
Denmark: 44186
Dominican Republic: 866
East Germany: 2095
Ecuador: 893
Egypt: 3090
El Salvador: 167
Estonia: 4651
Ethiopia: 537
Faroe Islands: 97
Federal Republic of Yugoslavia: 2167
Finland: 48990
France: 467279
Gabon: 80
Georgia: 1846
Germany: 226808
Ghana: 655
Greece: 20305
Greenland: 90
Grenada: 79
Guatemala: 638
Guinea: 90
Haiti: 498
Honduras: 80
Hong Kong: 66567
Hungary: 19668
Iceland: 7839
India: 110215
Indonesia: 3931
Iran: 11762

Iraq: 1365
Ireland: 26506
Isle Of Man: 396
Israel: 16662
Italy: 288407
Jamaica: 807
Japan: 178558
Jordan: 1313
Kazakhstan: 2145
Kenya: 523
Korea: 90
Kosovo: 188
Kuwait: 273
Kyrgyzstan: 400
Laos: 289
Latvia: 2297
Lebanon: 1289
Liberia: 283
Libya: 520
Liechtenstein: 1011
Lithuania: 3772
Luxembourg: 11976
Macao: 175
Madagascar: 204
Malaysia: 1338
Mali: 220
Malta: 885
Martinique: 103
Mauritania: 515
Mexico: 35560
Micronesia: 85
Moldova: 95
Monaco: 189
Mongolia: 186
Montenegro: 439

Morocco: 4525
Namibia: 105
Nepal: 797
Netherlands: 47054
New Zealand: 14561
Nicaragua: 183
Nigeria: 494
North Korea: 508
Norway: 26476
Pakistan: 1728
Palestine: 1530
Panama: 903
Papua New Guinea: 295
Paraguay: 317
Peru: 2436
Philippines: 7882
Poland: 35543
Portugal: 15232
Puerto Rico: 1322
Qatar: 1036
Republic of Macedonia: 1515
Reunion: 38
Romania: 15739
Russia: 50757
Rwanda: 527
Samoa: 110
Saudi Arabia: 490
Senegal: 1462
Serbia: 4337
Serbia and Montenegro: 1314
Singapore: 4557
Slovakia: 2525
Slovenia: 2956

Somalia: 90
South Africa: 14688
South Korea: 56245
Soviet Union: 51353
Spain: 116651
Sri Lanka: 318
Suriname: 105
Sweden: 68956
Switzerland: 34516
Syria: 447
Taiwan: 16618
Tajikistan: 351
Tanzania: 441
Thailand: 10189
The Democratic Republic Of Congo: 98
Trinidad and Tobago: 86
Tunisia: 2236
Turkey: 20068
UK: 492616
USA: 2276142
Uganda: 243
Ukraine: 4616
United Arab Emirates: 3202
Uruguay: 2038
Uzbekistan: 324
Vanuatu: 100
Venezuela: 2014
Vietnam: 1261
West Germany: 58480
Yugoslavia: 8742
Zaire: 80
Zimbabwe: 210

Task 2 (movies per year and genre) produces the following results:

1973_Adventure: 1
1973_Drama: 1
1973_History: 1
1976_Comedy: 2
1976_Drama: 1
1976_History: 1
1976_Mystery: 1
1976_Romance: 1

1979_Adventure: 1
1979_Crime: 1
1979_Mystery: 1
1982_Comedy: 1
1982_Drama: 1
1985_Documentary: 1
1985_History: 1
1988_Comedy: 1

1988_Drama: 2
1988_Sci-Fi: 1
1996_Documentary: 1
1996_Music: 1
2001_Documentary: 1
2001_Music: 1
2002_Documentary: 1
2002_Music: 1

2004_Adventure: 1
2004_Documentary: 2
2004_Drama: 1
2004_Music: 1
2004_Romance: 1
2006_Animation: 1
2006_Crime: 1
2006_Drama: 1

2007_Documentary: 1	2008_Crime: 1	2015_Action: 1
2007_War: 1	2008_Drama: 1	2015_Adventure: 1
2008_Action: 1	2009_Documentary: 1	2015_Animation: 1

C   Problem 3 execution results

We list here the execution results of our solution for the third problem (mining DNA sequence patterns).

AA: 328236	ACTA: 6151	ATGC: 20429	CCCG: 15287	CTAT: 8656	GCA: 92010	GGTC: 12705
AAA: 104317	ACTC: 9238	ATGG: 18458	CCCT: 8538	CTC: 40893	GCAA: 25483	GGTG: 21949
AAAA: 33151	ACTG: 19234	ATGT: 13456	CCG: 83289	CTCA: 11330	GCAC: 17151	GGTT: 19244
AAAC: 23753	ACTT: 12409	ATT: 79882	CCGA: 14961	CTCC: 9005	GCAG: 27535	GT: 248313
AAAG: 21525	AG: 231133	ATTA: 18033	CCGC: 27044	CTCG: 9953	GCAT: 20449	GTA: 50415
AAAT: 24342	AGA: 54161	ATTC: 16435	CCGG: 22950	CTCT: 9969	GCC: 88989	GTAA: 17001
AAC: 79028	AGAA: 17403	ATTG: 19826	CCGT: 17149	CTG: 98461	GCCA: 29986	GTAC: 11373
AACA: 20648	AGAC: 10050	ATTT: 24372	CCT: 48210	CTGA: 23008	GCCC: 14823	GTAG: 8902
AACC: 19257	AGAG: 10211	CA: 316026	CCTA: 3855	CTGC: 26589	GCCG: 26984	GTAT: 12435
AACG: 22994	AGAT: 15668	CAA: 73266	CCTC: 7577	CTGG: 31803	GCCT: 15890	GTC: 51835
AACT: 14967	AGC: 77456	CAAA: 22726	CCTG: 23291	CTGT: 15584	GCG: 109836	GTCA: 17297
AAG: 60678	AGCA: 21971	CAAC: 20654	CCTT: 12788	CTT: 60920	GCGA: 23903	GTCC: 7788
AAGA: 16071	AGCC: 16675	CAAG: 9104	CG: 336756	CTTA: 9264	GCGC: 33151	GTCG: 16319
AAGC: 18678	AGCG: 25147	CAAT: 19716	CGA: 67889	CTTC: 19829	GCGG: 26823	GTCT: 9694
AAGG: 12858	AGCT: 12547	CAC: 63948	CGAA: 17983	CTTG: 9033	GCGT: 24346	GTG: 63314
AAGT: 12151	AGG: 48468	CACA: 12311	CGAC: 16249	CTTT: 21881	GCT: 76778	GTGA: 17966
AAT: 79476	AGGA: 10559	CACC: 22156	CGAG: 9748	GA: 259534	GCTA: 10018	GTGC: 16924
AATA: 19820	AGGC: 15860	CACG: 15464	CGAT: 22883	GAA: 79933	GCTC: 12590	GTGG: 17237
AATC: 19663	AGGG: 8625	CACT: 13051	CGC: 110789	GAAA: 25923	GCTG: 34350	GTGT: 10240
AATG: 20320	AGGT: 12704	CAG: 100434	CGCA: 24078	GAAC: 16861	GCTT: 18657	GTT: 79059
AATT: 18529	AGT: 47616	CAGA: 20813	CGCC: 33151	GAAG: 19665	GG: 262488	GTTA: 16434
AC: 249408	AGTA: 9965	CAGC: 35398	CGCG: 26624	GAAT: 16328	GGA: 53753	GTTC: 17014
ACA: 56161	AGTC: 8810	CAGG: 23383	CGCT: 25227	GAC: 52428	GGAA: 19116	GTTG: 20618
ACAA: 15670	AGTG: 12816	CAGT: 19337	CGG: 83179	GACA: 12301	GGAC: 7800	GTTT: 23816
ACAC: 10371	AGTT: 15280	CAT: 73632	CGGA: 16056	GACC: 13006	GGAG: 9031	TA: 205922
ACAG: 15775	AT: 300819	CATA: 12645	CGGC: 27287	GACG: 17325	GGAT: 16986	TAA: 65882
ACAT: 13477	ATA: 60954	CATC: 24906	CGGG: 15314	GACT: 8996	GGC: 88242	TAAA: 21016
ACC: 71689	ATAA: 20773	CATG: 14370	CGGT: 23289	GAG: 40596	GGCA: 25961	TAAC: 16559
ACCA: 22611	ATAC: 12580	CATT: 20651	CGT: 70033	GAGA: 11040	GGCC: 11862	TAAG: 9452
ACCC: 11515	ATAG: 8878	CC: 263858	CGTA: 13496	GAGC: 12222	GGCG: 32556	TAAT: 17886
ACCG: 23794	ATAT: 17794	CCA: 82832	CGTC: 17128	GAGG: 7602	GGCT: 16561	TAC: 50347
ACCT: 12668	ATC: 82787	CCAA: 12835	CGTG: 15524	GAGT: 9126	GGG: 45460	TACA: 10059
ACG: 70094	ATCA: 26028	CCAC: 17548	CGTT: 22855	GAT: 82717	GGGA: 10600	TACC: 16192
ACGA: 13381	ATCC: 17273	CCAG: 32327	CT: 229235	GATA: 19089	GGGC: 14376	TACG: 13286
ACGC: 24945	ATCG: 22966	CCAT: 18863	CTA: 25613	GATC: 17963	GGGG: 8241	TACT: 10026
ACGG: 17017	ATCT: 15258	CCC: 45696	CTAA: 7124	GATG: 24737	GGGT: 11534	TAG: 26041
ACGT: 13674	ATG: 72908	CCCA: 12842	CTAC: 8601	GATT: 19727	GGT: 71145	TAGA: 5424
ACT: 47752	ATGA: 19462	CCCC: 8337	CTAG: 832	GC: 373030	GGTA: 16193	TAGC: 9993

TAGG: 3925	TCAC: 17939	TCGC: 24051	TGAA: 24226	TGGA: 15772	TTA: 65815	TTG: 73641
TAGT: 6296	TCAG: 23293	TCGG: 15163	TGAC: 17537	TGGC: 29388	TTAA: 19955	TTGA: 18260
TAT: 60595	TCAT: 19783	TCGT: 13856	TGAG: 11019	TGGG: 12603	TTAC: 17042	TTGC: 25756
TATA: 8542	TCC: 53611	TCT: 53062	TGAT: 25925	TGGT: 22534	TTAG: 7013	TTGG: 12823
TATC: 18970	TCCA: 16156	TCTA: 5213	TGC: 91073	TGT: 55861	TTAT: 20781	TTGT: 15705
TATG: 12412	TCCC: 10346	TCTC: 10881	TGCA: 18677	TGTA: 9995	TTC: 80237	TTT: 105118
TATT: 19788	TCCG: 15949	TCTG: 20097	TGCC: 25972	TGTC: 12434	TTCA: 24566	TTTA: 21113
TC: 259648	TCCT: 10428	TCTT: 16105	TGCG: 23857	TGTG: 12072	TTCC: 18731	TTTC: 25763
TCA: 80436	TCG: 68674	TG: 312973	TGCT: 21277	TGTT: 20499	TTCG: 18383	TTTG: 23085
TCAA: 18194	TCGA: 14613	TGA: 79892	TGG: 81523	TT: 329732	TTCT: 17327	TTTT: 33542

C.1   Problem 4 execution results

In the following 4-columned pages, the experimental results of section's 5 problem are presented, as the raw output of the 3rd MapReduce phase execution.

102: 20.772281	136: 13.721149	214: 19.809753000000004	260: 33.483572999999986
104: 29.609879	137: 13.660803000000001	215: 22.834134999999996	261: 32.508742999999996
105: 27.598046999999998	138: 13.738254000000001	216: 22.450407999999996	264: 12.784288
107: 14.800885000000003	140: 11.448863	217: 12.86093	265: 17.312394
109: 27.163701999999994	141: 12.826131	219: 13.857943	266: 16.947997
110: 16.304225	143: 12.803862	220: 26.279680999999997	267: 26.82996199999999
111: 42.771938	154: 13.723290000000002	221: 15.366022000000003	268: 13.143828000000001
112: 9.670065000000001	170: 35.83375399999999	222: 10.174368	269: 9.54012
113: 28.587929000000003	174: 39.935224000000005	223: 31.783763000000004	27: 18.650638999999998
114: 17.573652999999997	175: 42.11639699999999	226: 11.464125999999998	270: 12.200784000000002
115: 33.936378	176: 33.780936999999994	227: 9.910114	272: 27.8627
116: 25.765458000000002	180: 9.111828999999998	228: 26.034533	273: 20.449326000000003
117: 30.000875	187: 10.061923	229: 17.984762	274: 21.781969000000004
1180: 9.566474	191: 21.271617	23: 8.068862	275: 18.921694000000002
1181: 8.554887	192: 22.415023999999995	231: 11.600534	277: 13.899377000000001
1182: 10.499468	193: 24.815881999999995	232: 21.044421999999994	278: 45.28685699999998
1183: 8.272158999999998	194: 15.367017	24: 15.912360000000001	279: 36.933221999999994
1186: 8.551493	197: 19.721797	244: 20.739924000000002	280: 36.45238099999999
120: 8.667856	198: 20.10406	245: 26.42594699999999	283: 21.347036
121: 24.678666000000003	200: 26.785185	246: 12.422665	285: 39.845339
122: 62.881582	202: 20.554610999999998	247: 12.198811000000001	287: 15.863291
124: 23.14891	203: 26.437709999999992	249: 19.911965999999996	288: 24.174710999999995
128: 13.860000000000001	204: 21.193861000000002	25: 9.950251999999999	289: 20.507460000000005
129: 13.828262	205: 18.054205999999997	250: 20.231632999999995	29: 10.500267
130: 12.778032	206: 12.843962	251: 35.69873999999999	290: 15.802384000000002
131: 14.548629	208: 18.285626	252: 13.855876000000002	292: 34.518677
132: 13.780112000000003	209: 9.478037	254: 8.187458	293: 42.414617
133: 12.777426	210: 18.463774999999995	255: 27.296618	294: 39.387104
134: 17.525341	211: 26.206232	257: 20.167998999999995	295: 33.319982999999986
135: 12.858882000000001	213: 8.21817	258: 13.343784000000001	296: 26.437148

30: 13.124654000000001
300: 16.612497
301: 35.508855000000004
302: 40.46924
303: 30.324057999999997
304: 34.186174
306: 12.180608
308: 49.780783000000014
309: 21.140049
310: 16.898876
311: 9.782459
318: 11.424189000000002
32: 7.906732000000001
320: 15.995923000000003
322: 12.928547
323: 12.95803
324: 14.145467000000002
325: 16.820611
326: 9.210550999999999
327: 8.77006
328: 11.502478
329: 9.477534
330: 7.819681999999999
331: 12.199909
333: 17.355712
334: 16.773555
335: 12.816709
336: 9.472820000000002
337: 16.717836
338: 14.435681000000002
34: 8.902532
341: 13.712019000000002
342: 12.777168000000001
343: 21.60238899999999
344: 13.763916
345: 9.50445
346: 13.803495
347: 13.770425000000001
348: 13.696388000000002
349: 12.777321
35: 8.831461000000001
350: 13.700356000000001
351: 12.82703
352: 18.913772999999996
353: 13.840302000000001
354: 12.026345

355: 13.838998000000002
356: 13.709090000000002
36: 7.751157000000001
361: 13.740885
37: 9.843972
378: 7.955433
379: 11.753690999999998
38: 10.698117
380: 9.03648
381: 31.255191
385: 16.697326
386: 13.088604999999998
389: 43.11891100000001
39: 10.792126000000001
391: 7.855742000000001
40: 7.842008000000001
405: 18.636048
406: 30.94102599999999
408: 16.025723
409: 21.124420999999998
410: 19.690642
412: 12.448818000000001
413: 21.007549999999995
414: 19.201738999999996
415: 20.822085
416: 20.244446999999994
417: 20.336876999999998
418: 20.905703999999997
419: 16.571078
420: 14.833722000000003
421: 22.127405999999997
422: 20.207241999999997
423: 19.837138999999997
424: 22.204759999999993
425: 20.3222
427: 18.573423000000002
428: 18.402874999999998
429: 20.996679999999998
431: 17.247738000000005
432: 7.730989
433: 20.990592000000003
437: 13.467927000000001
438: 22.538098999999995
439: 20.257774999999995
440: 18.640948999999996
441: 22.565751999999993

442: 22.468430999999992
443: 22.236632999999994
444: 12.668758
445: 21.391153999999997
446: 20.229419
447: 23.098830999999993
448: 23.389274999999994
449: 21.425622999999995
450: 20.304112999999994
451: 21.309653999999995
452: 21.574356999999992
453: 21.216737999999992
454: 21.305387999999994
455: 22.443621999999994
456: 18.427684
457: 23.612626999999996
458: 20.137554999999995
468: 16.434241
473: 15.624302000000002
490: 15.118589000000002
492: 29.037201999999994
494: 14.964478999999999
497: 7.779587000000001
505: 22.493396999999998
507: 16.502023
508: 13.205226000000001
509: 10.297368000000002
510: 17.989523
511: 14.096020000000003
512: 7.909458999999998
529: 16.491923000000003
530: 20.54695299999999
531: 13.581469
532: 18.63035
533: 16.137291
534: 15.585291000000002
535: 13.489995
536: 15.663219000000002
537: 15.558095000000002
538: 20.604023999999995
539: 15.567038000000002
540: 10.592666000000001
541: 13.301682
542: 13.708515000000002
544: 8.796812000000001
545: 7.992185

546: 8.689068
547: 17.435207
548: 14.626078000000001
549: 8.902671000000002
550: 15.475738000000002
563: 12.416060000000002
604: 14.693641000000001
605: 14.628854
606: 13.673755000000003
607: 13.823655000000002
608: 13.666609
609: 14.749676000000001
610: 14.726731000000003
611: 12.480203
612: 21.530917999999996
639: 16.345097000000003
64: 11.651773
640: 26.79545999999999
641: 27.95541699999999
642: 22.714243999999994
643: 28.925386
644: 27.951106
645: 9.891883
646: 22.266540999999997
647: 17.500691000000003
653: 31.20108699999998
654: 11.212689
655: 15.470210000000002
656: 13.714576000000001
659: 14.530663000000002
660: 14.484713000000001
661: 14.486255000000002
663: 14.393031
664: 15.657084000000003
665: 12.470333000000002
666: 15.570483000000001
667: 8.476292
668: 8.400375
669: 12.719472000000001
670: 12.759070000000001
671: 18.786868000000002
672: 12.737939000000003
673: 11.354866
674: 12.464726000000002
675: 14.512069
676: 12.633483

677: 15.30606
678: 13.630705
679: 22.209649
680: 26.49534299999999
681: 11.586703
682: 11.780345000000002
683: 10.837639
684: 11.369396
685: 20.680481999999994
686: 28.644553999999992
687: 13.618457000000001
70: 12.576885000000003
702: 15.804968
703: 18.989590999999997
711: 8.769808000000001
72: 13.937429
729: 11.920043
73: 33.317902999999994
739: 14.833940000000002
74: 9.418914
741: 17.572777
743: 16.54965
744: 13.711973000000002
745: 14.759685000000001
746: 14.843160000000001
75: 8.17184
76: 35.16477499999998

765: 8.972965
773: 9.301499
778: 9.388625000000001
779: 8.382993
78: 35.074965
79: 18.829224
792: 9.650819
793: 21.405161999999994
794: 7.747058000000001
796: 7.746258000000001
797: 20.176615999999996
80: 19.708188999999997
81: 27.764969999999987
82: 8.954957
822: 7.824752000000001
827: 26.284926
828: 31.25722999999999
83: 16.828261
830: 12.777338
84: 17.39505
845: 19.799999999999997
846: 23.372872999999995
847: 21.634598999999994
848: 21.599697999999993
849: 20.692736999999997
85: 17.810776
850: 21.486940999999995

851: 20.637819999999998
852: 20.758516999999994
853: 19.714644999999994
854: 20.674989999999998
858: 10.127550999999999
859: 13.744827
86: 9.620828000000001
860: 13.392533000000002
861: 12.716919
862: 14.757012000000001
863: 14.737124000000001
870: 12.684738000000001
871: 12.674369
872: 11.814398
873: 12.763509
874: 10.89
875: 12.642457000000002
876: 12.794634000000002
877: 9.48321
878: 12.762451
879: 13.635698000000001
880: 12.768133
881: 12.801868000000002
882: 10.89
893: 7.887907
894: 9.918199
895: 16.402685000000005

896: 15.014125
901: 10.018733
902: 18.710283
915: 9.631426000000001
917: 9.638277000000002
918: 10.553774
919: 8.431745
922: 10.480677000000002
926: 20.516713999999997
927: 18.237036999999997
928: 18.419195
929: 20.752049999999997
930: 13.043696999999998
931: 18.611175
932: 20.685094999999997
933: 20.786459999999995
935: 20.676130999999998
936: 21.667931999999997
937: 19.520974999999996
941: 25.234674999999996
948: 16.893155
950: 20.797809999999995
951: 19.461768999999997
953: 18.305716999999998