The Map-Reduce Programming Paradigm

M.Sc. course on "Technologies for Big Data Analysis" - Assignment 2

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

1 Introduction

The current document is a technical report for the second 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 2 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 acquainted with the MapReduce programming model in Hadoop framework in Java programming language. This is a WordCount problem's variation.

2.1 Problem Statement

Implement a **MapReduce** program, a variation of the **word-count** problem, to generate and count "numeronyms". Numeronyms correspond to words as shown below:

- s5n – shorten
- h7k – hyperlink
- l10n – localization

[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.

- i18n – internationalization

A numeronym of a word is defined as an alphanumeric string formed by taking the first and last character of the word and inserting between them the count of characters between the first and the last. More specifically, the program should process words with a length of 3 characters or more, generate numeronyms, and then output the frequency of each numeronym.

The program should ignore:

- words shorter than 3 characters
- punctuation marks
- uppercase/lowercase differences, i.e., it should be *case-insensitive*

Additionally, a parameter k will be given, representing the minimum number of occurrences of a numeronym that we are interested in. The program output should be a list of numeronyms and the frequency of each numeronym that is greater than or equal to the parameter k specified by the user.

2.2 Proposed approach

2.2.1 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 `NumeronymsMaster` 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/ numeronyms /input`.
- `<output_path>` specifies the directory where the MapReduce output will be written, which will be created in the `out` folder.
- <k> is the minimum frequency threshold for numeronyms 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.

2.2.2 Implementation. The proposed approach leverages the MapReduce programming model to compute numeronyms from a given input dataset. A numeronym is a concise representation of a word, typically formed by retaining the first and last letters while replacing the intervening characters with their count. This section outlines the methodology implemented across three key components: the driver class, the mapper, and the reducer. This methodology supports flexible configuration through command-line arguments, enabling users to specify the threshold k and the minimum word length for numeronym generation. The use of the MapReduce paradigm ensures scalability and parallel processing for large datasets, making the approach well-suited for distributed systems.

The driver class, `NumeronymsMaster`, orchestrates the overall MapReduce job. It takes command-line arguments for the input and output directories as well as a threshold parameter k, which determines the minimum frequency for numeronyms to be included in the final output. The following steps are executed:

- Input and output paths are parsed, and any pre-existing output directory is deleted using the `Utilities.deleteDirectory` method.
- A Hadoop configuration object is initialized with two parameters:

  - `numeronyms.k`: The threshold k.
  - `numeronyms.l`: The minimum word length (default value is 3) required for numeronym generation.
- The job is configured to use the `NumeronymMapper` and `NumeronymReducer` classes, with input and output formats set to `TextInputFormat` and `TextOutputFormat`, respectively.
- Finally, the job waits for completion before terminating.

The `NumeronymMapper` class is responsible for processing each line of the input text file and emitting numeronym -frequency pairs. The following steps are performed:

- During setup, the minimum word length (`numeronyms.l`) is retrieved from the configuration.
- Each input line is tokenized into individual words, and non-alphabetic characters are removed from each token.
- For valid words (those with lengths greater than or equal to the specified minimum), a numeronym is constructed by concatenating the first character, the number of intervening characters, and the last character.
- A key-value pair is emitted, where the key is the numeronym and the value is a count of one.

The `NumeronymReducer` class aggregates the frequency counts for each numeronym and filters out those below the specified threshold k. The process includes:

- Retrieving the threshold value (`numeronyms.k`) from the configuration during the setup phase.
- Summing the values for each numeronym key received from the mapper.
- Emitting numeronyms that meet or exceed the threshold k along with their corresponding frequencies.

2.2.3 Evaluation. The experiments were executed using the dataset 'SherlockHolmes.txt' with a parameter setting of $k = 10$. This configuration was chosen to evaluate the performance and the behavior of the system under moderate conditions. The dataset, containing a series of textual records, was processed to analyze various patterns and results.

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

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:

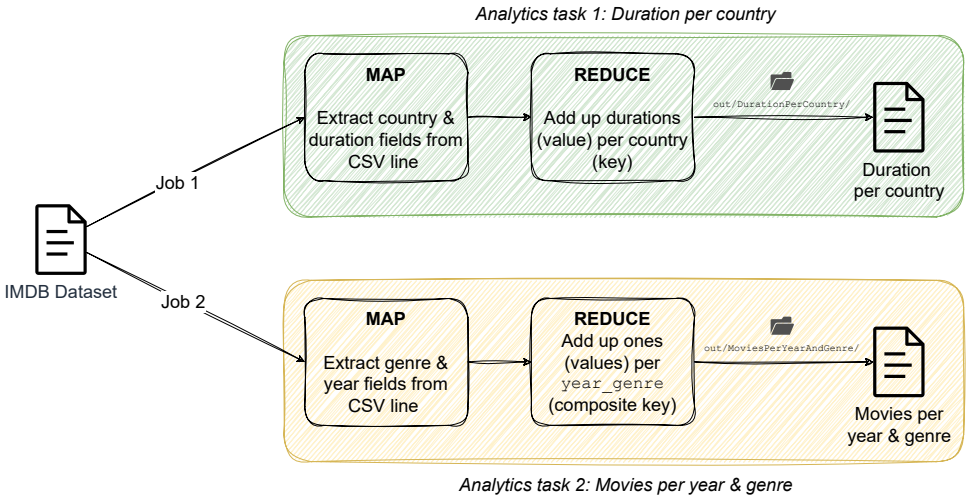


Fig. 1. The MapReduce solution architecture for the movie analytics problem. Two separate jobs are executed to solve the two tasks, namely duration per country (green) and movies per year & genre (yellow). The two tasks also create separate subdirectories for writing the results.

- 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.

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 1 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 1 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 1 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:

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

Algorithm 1 Populating DNA subsequences

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

- `<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.

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

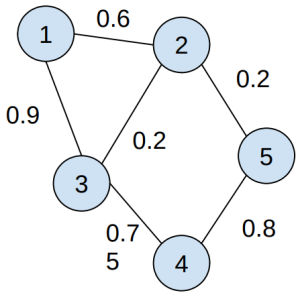
We are given an input file in text format where each line contains a connection between two vertices of a network and a probability value. For each edge e, there is a probability value $p(e)$ which indicates the probability that the two vertices are connected by the edge. Obviously, the values of $p(e)$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

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

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

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

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

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

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

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