Data Clustering Contest: Round 1

> General info about the contest is available on @contest. See also: This page in Russian

The task for this contest is to create a grouping and sorting algorithm for news articles.

Source Data

A test data set is available in HTML format:

Sample 1, 01-07
Sample 2, 08-17
Sample 3, 18-21
Sample 4, 22-25

We will publish additional data sets during the course of the contest.

> We will use a different data set for evaluating submissions. The evaluation data set may also include articles from domains not present in the test data set.

The Task

1. **Isolate articles in English and Russian.** Your algorithm must sort articles by language, filtering English and Russian articles. Articles in other languages are not relevant for this stage of the contest and may be discarded.
2. **Isolate news articles.** Your algorithm must discard everything except for news articles.
3. **Group news articles by category.** Your algorithm must place news articles into the following **7** categories:
    - Society (includes Politics, Elections, Legislation, Incidents, Crime)
    - Economy (includes Markets, Finance, Business)

- Entertainment (includes Movies, Music, Games, Books, Arts)
- Science (includes Health, Biology, Physics, Genetics)
- Other (news articles that don't fall into any of the above categories)

4. **Group similar news into threads.** Your algorithm must identify news articles about the same event and group them together into threads, selecting a relevant title for each thread. News articles inside each thread must be sorted according to their relevance (most relevant at the top).
5. **Sort threads by their relative importance.** Your algorithm must sort news threads in each of the categories based on perceived importance (important at the top). In addition, the algorithm must build a global list of threads, indepedent of category, sorted by perceived importance (important at the top).

Submitting Your Work

You must submit a standalone app with the name **tgnews** and a command line interface as described below. During evaluation, the app will be launched with the following parameters:

```
tgnews languages source_dir
tgnews news source_dir
tgnews categories source_dir
tgnews threads source_dir
tgnews top source_dir
```

source_dir – path to the folder with HTML-files containing article texts.

Requirements

Your app must work locally (no network usage).
Speed is of utmost importance (this may give an edge to apps written in **C++**).
External dependencies should be kept to a minimum. If you can't avoid external dependencies, please list them in a text file named **deb-packages.txt**. These dependencies will be installed using `sudo apt-get install ...` before your app is tested.

We will not evaluate apps that require more than **60** seconds for each batch of **1000** files passed in source_dir. This restriction must be met for each of the 5 script launches during evaluation.

You must submit a ZIP-file (submissions that exceed 200 MB are likely to be penalized) with the following structure:

```
submission.zip
  -> tgnews - executable binary file with an interface as described below
  -> src - folder with the app's source code
  -> deb-packages.txt - a text file with line-break separated debian package names of all external depende
  -> * - any additional resources your app requires to work (please use relative paths to access them)
```

Evaluation Process

Each of the submitted apps will be checked in several stages. The app will be launched several times with different parameters described below.

> We expect the algorithm to be able to identify and work with articles in English and Russian.

1. Isolate articles in English and Russian

Launch parameters:

```
tgnews languages source_dir
```

The result must be output to STDOUT in JSON format.

Output format:

```
[
  {
    "lang_code": "en",
    "articles": [
      "981787246124324.html",
      "239748235923753.html",
      ...
    "lang_code": "ru",
    "articles": [
      "273612748127432.html",
      ...
    ]
  },
  ...
]
```

where:

> **lang_code** – ISO 639-1 two-letter language code
> **articles** – list of file names containing texts in **lang_code** language

2. Isolate news articles

Launch parameters:

```
tgnews news source_dir
```

The result must be output to STDOUT in JSON format.

Output format:

```
{
  "articles": [
    "981787246124324.html",
    ...
  ]
}
```

where:

> **articles** – list of file names containing news

3. Group news articles by category

The result must be output to STDOUT in JSON format.

Output format:

```
[
  {
    "category": "society",
    "articles": [
      "981787246124324.html",
      ...
    ]
  },
  {
    "category": "sports",
    "articles": [
      "2348972396239813.html",
      ...
    ]
  },
  ...
]
```

where:

- **category** – `"society"` , `"economy"` , `"technology"` , `"sports"` , `"entertainment"` , `"science"` or `"other"`
- **articles** – list of file names containing news articles that match the **category**

4. Group similar news into threads

Launch parameters:

```
tgnews threads source_dir
```

The result must be output to STDOUT in JSON format.

```
  {
    "title": "Telegram announced Data Clustering Contest",
    "articles": [
      "6354183719539252.html",
      ...
    ]
  },
  {
    "title": "Apple reveals new AirPods Pro",
    "articles": [
      "9436743547232134.html",
      ...
    ]
  },
  ...
]
```

where:

- **title** – thread title
- **articles** – list of file names containing articles in the thread, sorted by their relevance (most relevant at the top)

5. Sort threads by their relative importance

Launch parameters:

```
tgnews top source_dir
```

The result must be output to STDOUT in JSON format.

Output format:

```
[
  {
    "category": "any",
    "threads": [
        "articles": [
          "6354183719539252.html",
          ...
        ]
      },
      {
        "title": "Apple reveals new AirPods Pro",
        "category": "technology",
        "articles": [
          "9436743547232134.html",
          ...
        ]
      },
      ...
    ]
  },
  {
    "category": "technology",
    "threads": [
      {
        "title": "Telegram announces Data Clustering Contest",
        "articles": [
          "6354183719539252.html",
          ...
        ]
      },
      ...
    ]
  },
  {
    "category": "sports",
    "threads": [
      ...
    ]
  },
```

where:

category – `"society"` , `"economy"` , `"technology"` , `"sports"` , `"entertainment"` , `"science"` or `"other"` . You must also include a block with `category="any"` , containing a list of threads, indepedent of category, sorted by relevance (most relevant at the top).
threads – list of threads, sorted by relevance (most relevant at the top). Each thread contains a title and list of articles. If `category="any"` , also includes **category**.
title – thread title.
articles – list of names of files containing articles from the thread, sorted by relevance (most relevant at the top).

Clarifications

The recommended size for your resulting archive is below 200 MB. You can submit an archive of up to 1,5 GB but submissions exceeding 200 MB are likely to be penalized during evaluation.
The 60 seconds per 1000 files restriction must be met for each of the 5 script launches during evaluation.