INFO2 – Micro Project

Media Transparency Austria

Objectives

Write a command-line application that allows to run simple queries against a quarterly published data set on media transparency

Background

As a consequence of ongoing discussions about potential misuse of tax-payers' money spent in advertising, the Austrian Parliament has released the so called Media Transparency Act ("Bundesverfassungsgesetz über die Transparenz von Medienkooperationen sowie von Werbeaufträgen und Förderungen an Medieninhaber eines periodischen Mediums und das KommAustria-Gesetz") in 2011. This regulation requires certain organizations to quarterly report on their spending in advertising and funding of media cooperations. Reported data is published by the Rundfunk und Telekom Regulierungs GesmbH (RTR).

These data can be downloaded from RTR's web site in different formats. We are going to use the REST-based JSON interface.

Grading

This micro project is structured into features and feature sets. Every feature is graded individually. Thus, you do not need to implement all features in order to get some points. Partial implementations are OK as well. The total number of points depends on the number of correctly implemented features and some none-functional features of your code like readability and performance (please see the evaluation sheet for more details).

The project can be done in groups of two or alone. If you are doing it alone, you qualify for a 20% surplus in points (but generally cannot exceed the maximum of points).

There will be submission interviews with each team individually in which you are required to explain your code and will probably be asked to perform minor improvements/modifications on the fly! Please bring the filled in evaluation sheet to the interview!

Submission & Deadline

One solution per team needs to be uploaded to moodle (.hs file only). This has to happen **before June 17th, 2019, 8:00am**.

Feature Sets

Here are the requirements structured in features or feature sets

Feature Set 0 - Program Control Loop

When your program got started, it should ask the user for a command to enter:

```
Enter your command or type 'help' for assistance
```

If the user types in `help`, a list of all supported (=actually implemented) commands is displayed:

```
 'help' ... print this message
 'exit' ... terminate this application
 'payers' ... print a list of all payers
 'recipients' ... print a list of all recipient
 'top' n 'payers'|'recipients' '§2'|'§4'|'§31' ... print the n biggest payers|
    recipients for the given payment type
 'search' 'payers'|'recipients' searchTerm ... print a list of all payers|
    recipients containing the given searchTerm
 'load' quarter1 quarter2 .. quartern ... load date for the given list of
    quarters
 'details' 'payers'|'recipients' organization ... print a list of all payments
     payed or received by the given payers/recipient
```

If the user types in `exit`, the program prints `Bye!` and terminates. In case that the user types in a command that is not supported by the program, the following message appears:

```
Enter your command or type 'help' for assistance
test
Sorry, but 'test' is unknown!

Enter your command or type 'help' for assistance
```

Feature Set 1 - Loading Data via the Web

Once the application got started, it downloads the data from the RTR web site. The URL is the following:

```
https://data.rtr.at/api/v1/tables/MedKFTGBekanntgabe.json?
quartal=[Quarter]&leermeldung=0&size=0
```

`[Quarter]` is a placeholder for the quarter the user is interested in. It always has the form `yyyyq`. Thus, all quarters of 2018 look as follows:

- 20181
- 20182
- 20183
- 20184

The data comes in JSON format and looks as follows:

```
{
  "message": "OK",
  "status": 200,
  "timestamp": "2019-05-14T12:09:02+02:00",
  "data": [
    {
      "rechtstraeger": "FH JOANNEUM Gesellschaft mbH",
      "quartal": "20184",
      "bekanntgabe": 2,
```

```
      "leermeldung": 0,
      "mediumMedieninhaber": "Kleine Zeitung",
      "euro": 5439.18
    }
  ],
  "version": {
    "id": 13281,
    "published": "2019-03-14T11:13:00+01:00"
  }
}
```

It is a JSON object with five properties. The information we are interested in is stored in property `data`, which in turn holds an array with all payment data (in this sample case, there's only one entry, though). Each entry in this array has the following properties:

Name	**Meaning**
rechtstraeger	This is the name of the company or organization that pays money
quartal	The quarter in which this transfer was made
bekanntgabe	The reason why this money was payed. The number refers to a paragraph in the law. §2 indicates advertising, while §4 covers funding. §31 covers fees received by the Austrian public broadcasting company (ORF)
leermeldung	Not important to us, since we only get non-empty records
medieninhaber	This is the name of the recipient of the money
euro	The total amount of money that was transferred in the given quarter from the payer to the recipient

If the program is started without any additional information, it automatically loads data for the most recent quarter (which is currently 20184). Once this data was successfully downloaded, a feedback message appears:

```
> media.exe
loading data for ["20184"]
loaded data for 20184

Enter your command or type 'help' for assistance
```

Besides this, it should be possible to provide quarters that should be downloaded via the command line. E.g.:

```
> media.exe 20174 20181 20182
loading data for ["20174","20181","20182"]
loaded data for 20181
loaded data for 20174
loaded data for 20182

Enter your command or type 'help' for assistance
```

In this case only those quarters that are listed by the user get downloaded. Corresponding confirmation messages occur once the data is available.

Feature Set 2 – Listing Participants

Using the command `payers` prints an alphabetically sorted list of all payers. Every payer occurs exactly once. E.g.:

```
Enter your command or type 'help' for assistance
payers

Abfallwirtschaft Tirol-Mitte Ges.mbH.
Abfallwirtschaftsverband Spittal/Drau
Achenseeschiffahrt-GesmbH
...
Dornbirner Tourismus & Stadtmarketing GmbH
easy green energy GmbH & Co KG
EBG MedAustron GmbH
ebswien hauptkläranlage Ges.m.b.H.
ecoplus.Niederösterreichs Wirtschaftsagentur GmbH
EDITEL Austria GmbH
EKZ Tulln Errichtungs GmbH
...
```

The command `recipients` does the same for the recipients:

```
Enter your command or type 'help' for assistance
recipients
"Die Presse" Verlags-Gesellschaft m.b.H. & Co KG
"Freier Rundfunk Salzburg", Verein zur Förderung von freien, lokalen Radio- und
Fernsehprojekten"
"Salzburger Nachrichten" Verlagsgesellschaft m.b.H. & Co KG
"Steirische Vereinigung für Menschen mit Behinderung (STVMB)"
"ÖSTERREICHISCHE NATURSCHUTZJUGEND" (önj) Bundesverband
110%
1424-Magazin
4M Digital Media OG
6020 Stadtmagazin
88.6 - Der Musiksender
88.6 - Der Musiksender (Mostviertel)
88.6 - Wir spielen was wir wollen
A la Carte
A. Digital Errichtungs- und Beteiligungs GmbH
a3BAU
a3ECO
Abgeltung gemäß § 31 Abs. 11 ORF-G
```

Feature Set 3 – Quarters Overview

The quarters command calculates an overview of all the currently loaded quarters by providing the total sum for every payment category (§2, §4 and §31) in list sorted by quarter:

```
Enter your command or type 'help' for assistance
quarters
20174,      57.198.136,32 (§2),      4.258.839,36 (§4),    157.499.996,16 (§31)
20181,      29.526.617,60 (§2),     19.838.453,76 (§4),    157.799.997,44 (§31)
20182,      46.632.089,60 (§2),      7.352.736,00 (§4),    157.700.003,84 (§31)

Enter your command or type 'help' for assistance
```

Feature Set 4 – Re-loading Data

The load command allows for changing the current data set. It takes a list of required quarters and downloads their data. Again there's a confirmation message, once a quarter was successfully downloaded:

```
Enter your command or type 'help' for assistance
load 20181 20182 20183 20184

loading data for ["20181","20182","20183","20184"]
loaded data for 20183
loaded data for 20181
loaded data for 20184
loaded data for 20182

Enter your command or type 'help' for assistance
```

Feature Set 5 – Top Payers/Recipients

This is the first command that allows us to get some more interesting information out of the data set. It will find the biggest players in the game.

The top command has the following form:

```
'top' x  ('payers'|'recipients') ('§2'|'§4'|'§31')
```

where x is the number of entries that shall be returned by the command. Thus, in order to find the 10 media corporations that make most money out of public advertising, we can use the following command:

```
Enter your command or type 'help' for assistance
top 10 recipients §2
Kronen Zeitung                                  :   17.845.323,52
Heute                                           :   11.871.421,44
ORF 2                                           :    8.491.960,32
Kurier                                          :    6.851.409,28
Kleine Zeitung                                  :    5.445.127,68
Österreich                                      :    4.759.119,36
Österreich - oe24                               :    4.322.008,96
Die Presse                                      :    4.301.895,40
Der Standard                                    :    4.301.708,16
www.google.at                                   :    3.306.600,96
```

In order to find the five biggest spenders that are granting funds to media corporations we can use the top following in the following form:

```
Enter your command or type 'help' for assistance
top 5 payers §4
Rundfunk und Telekom Regulierungs-GmbH (RTR-GmbH) :   18.948.259,84
Kommunikationsbehörde Austria ("KommAustria")     :    8.065.939,20
Stadt Wien                                        :    2.312.272,96
Land Steiermark                                   :    1.804.189,44
Bundeskanzleramt                                  :    1.425.160,16
```

Feature Set 6 – Search

With this command we can search the list of payers or recipients for specific entries. It's syntax has the following form:

```
'search' ('payers'|'recipients') name
```

Where `name` is the name of the organization we are looking for. For example we could search for payers who have 'joa' in their names:

```
Enter your command or type 'help' for assistance
search payers joa
FH JOANNEUM Gesellschaft mbH
JOANNEUM RESEARCH Forschungsgesellschaft mbH
Universalmuseum Joanneum GmbH

Enter your command or type 'help' for assistance
```

Note: The search is case-insensitive!

Of course, we can also search for recipients:

```
Enter your command or type 'help' for assistance
search recipients kleine zei
Kleine Zeitung
Kleine Zeitung GmbH & Co KG
kleine Zeitung
```

Feature Set 7 – Details

This feature allows us to analyze the flow of payments for a specific payer or recipient. The syntax of the search command has the following form:

```
'details' ('payers'|'recipients') name
```

Here's an example that allows us to see all the payments made by our university:

```
Enter your command or type 'help' for assistance
details payers FH JOANNEUM Gesellschaft mbH
```

```
Payments according to §2:
Kleine Zeitung                                                  55.651,18
Der Standard                                                    19.886,00
www.google.at                                                   16.388,00
Die Presse                                                      13.168,00
Kronen Zeitung                                                  10.745,00
Radio Soundportal                                                8.262,00
Steiermark Magazin                                               6.900,00
Antenne Steiermark                                               6.050,00

Payments according to §4:

Payments according to §31:

Enter your command or type 'help' for assistance
```

It prints all money payed (or received) by payment type (§2, §4 and $31). Entries are sorted by amount in descending order.

Necessary Info

You can use any Haskell library you want. If you do so, please list all libraries that need to be installed additionally in a comment at the very beginning your file along we the names of the team members. E.g.

```
{-
Team: Student 1, Student 3
additional Libraries: aeson
-}
```

You can freely choose a name for your program file.

Hints

There is a very efficient library for turning raw JSON data into Haskell data types that is called aeson. You can easily add it to your programming infrastructure by running `cabal install aeson`.

In order to produce some nicely formatted output `Text.Printf` might turn out to be helpful!