

Ein Bild, das Axt, Vektorgrafiken, Werkzeug enthält.

Automatisch generierte Beschreibung

fetch data from twitter for   
1. deriving sentiment of each tweet   
2. getting top 10 hash tags / users  
3. retrieving followers of a given twitter user  
4. retrieving Tweets and profiles of followers of a given twitter user

Twitter Data Analysis

*OTH Regensburg  
Applied Data Science with Python*

Content

[1. Introduction 3](#_Toc104800452)

[2. The Project 3](#_Toc104800453)

[3. Tools, Modules and Data Structures 4](#_Toc104800454)

[3.1 Setup / Tools 4](#_Toc104800455)

[3.2 Modules 4](#_Toc104800456)

[3.3 Data Structures 4](#_Toc104800457)

[3.4 Twitter Keys/Tokens 6](#_Toc104800458)

[3.5 User Interface 6](#_Toc104800459)

[4. Class Diagram 7](#_Toc104800460)

[5. Listing of necessary functions for each Task 9](#_Toc104800461)

[5.1 Methods for Task 1 9](#_Toc104800462)

[5.2 Methods for Task 2 9](#_Toc104800463)

[5.3 Methods for Task 3 9](#_Toc104800464)

[5.4 Methods for Task 4 9](#_Toc104800465)

[6. Program Sequence 10](#_Toc104800466)

[6.1 Logic behind getting DataFrames 10](#_Toc104800467)

[6.1.1 DataFrame Type 1 10](#_Toc104800468)

[6.1.2 DataFrame Type 2 10](#_Toc104800469)

[6.2 Menu 12](#_Toc104800470)

[6.2.1 Main Menu 12](#_Toc104800471)

[6.2.2 Browse Tweets/Users (Submenu 0) 13](#_Toc104800472)

[6.2.3 Sentiment Analysis (Submenu 1) 14](#_Toc104800473)

[6.2.4 Get Top 10 Users/Hashtags (Submenu 2) 15](#_Toc104800474)

[6.2.5 Get Followers (Submenu 3) 16](#_Toc104800475)

[6.2.6 Obtain tweets and profiles of followers of given twitter user (Submenu 4) 17](#_Toc104800476)

[6.2.7 Change Topic 18](#_Toc104800477)

[7. Results 19](#_Toc104800478)

[8. Source Code / Instructions 19](#_Toc104800479)

[4.1 Source Code/Instructions 20](#_Toc104800480)

[4.3 Results 21](#_Toc104800481)

[4.4 Outlook 24](#_Toc104800482)

# Introduction

This document provides a report of the project “TwitterDataAnalyis” (see [2. The project](#_The_Project_1)) including following information:

* A brief description of the project
* Tools, Modules and Data Structures
* Class diagram
* Listing of necessary functions for each Task
* Program Sequence (flowcharts)

Moreover, the results of the project are described and enriched with screenshots for visualization.

Lastly the source code of the project and instructions how to use the project along with the requirements of my project are provided.

# The Project

The objectives of this project – twitter data analysis – is to fetch data from Twitter by utilizing the twitter API and provide the user requested data. Hereby fetched and filtered data by below criteria shall be provided/displayed to the user in a convenient and clear manner.

Following four Tasks must be considered:

1. Derive the sentiment of each tweet using Python module.  
2. Top 10 hash tags and users based on their number of tweets in your data set.  
3. Get the followers of a given twitter user from your acquired data set.  
4. Given a twitter user, obtain the tweets and profiles of all followers of the user and show it.

# Tools, Modules and Data Structures

## 3.1 Setup / Tools

For the development of the project Visual Studio Code will be used as integrated development environment (IDE). Python – version 3.10.3 - will be the used programming language leveraged by installing the ‘Python’ and ‘Python for VSCode’ extensions in Visual Studio Code. Additionally, a virtual environment will be utilized to install all required libraries into and having the advantage of keeping the system clean.

As this project requires to utilize the Twitter API for fetching data from Twitter a developer app linked to my Twitter account was created. Also – almost needless to mention – the Twitter dashboard for the developer app will be used to observe the count of requests already made. Furthermore, it will be taken advantage of the tool Postman mainly for quickly testing API requests.

As versioning tool git and the network-based service GitHub will be used.

## 3.2 Modules

Python has a big advantage in terms of variety of libraries und therefor several python libraries will be used. These required libraries are:

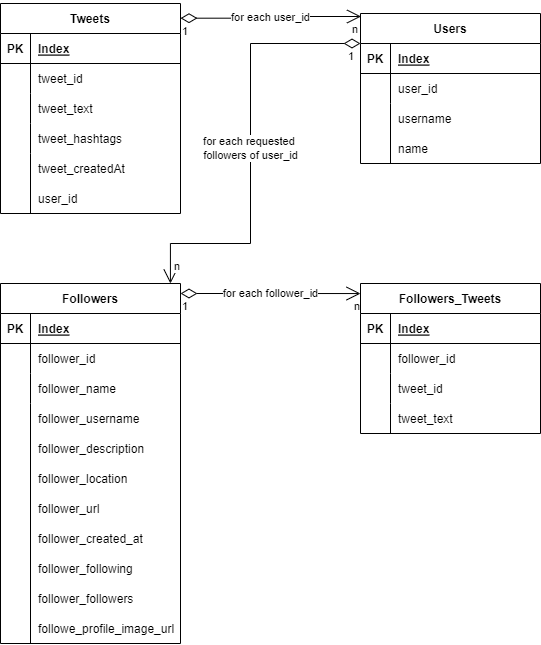
* os: to properly load environment variables, e.g. Access Keys for authentication to Twitter, into the program
* pandas: to process big data sets retrieved from Twitter API
* csv: to generate csv file/data structures and process those
* tweepy: for making API requests to the twitter api endpoint
* textblob: for text processing, particularly for sentiment analysis
* simple-term-menu: creating a easy to use and convenient user menu on the command line
* wordcloud: for creating wordcloud with mask of twitterlogo for most used words
* collections: for importing Counter and counting top 10 hashtags/users
* re (Regular Expressions): to remove eventually misleading URLs for analysing sentiment of Tweets
* pydoc: for automatically creating html documentation of classes/methods with docstrings
* black: Code formatter for unified formatting of python code

*Annotation:* Also have a look at the *requirements.txt* in the GitHub repository.

## 3.3 Data Structures

To temporarily store/load data and process available data sets, pandas DataFrames are used. Altogether four DataFrames are needed: One DataFrame for tweets, one for users, one for followers of a user and lastly one for tweets of followers of a given user.

Following diagram will clarify and visualize above written approach:



The ‘*Tweets*’ DataFrame contains the tweets fetched in the API requests with ‘tweepy’ and consists of 5 columns namely tweet\_id, tweet\_text, tweet\_hashtags, tweet\_createdAt and the user\_id. The latter is the author of the tweet. Tweet\_hashtags is a comma separated string of hashtags as loading DataFrames from csv files can cause a problem if elements are stored as lists. Tweet\_id is the ID of the tweet and tweet\_text is the corresponding tweet whereas tweet\_createdAt is the timestamp when the tweet was sent.

The ’*Users*’ DataFrame is somehow related to the Tweets DataFrame as for each user – exactly for each user\_id – the user is looked up and the username and name of the user is fetched via tweepy request. The user ID and corresponding username and name is stored within the Users DataFrame.

If the user wants to get the followers of a user ([Task 3](#Task3)), the ‘*Followers*’ DataFrame will be utilized to store the fetched followers of the desired user entered by the user of this program. This DataFrame also holds the data for the description – also known as bio or profile of a user - of followers. The user can get these profiles in another option of the main menu which partially fulfills [Task 4](#Task4).

If the user wants to get the profiles and tweets of all followers of entered user as part of [Task 4](#Task4), the last DataFrame ‘*Followers\_Tweets*’ is used to store all tweets fetched for all followers. Hereby this DataFrame consists of three columns namely follower\_id, tweet\_id and tweet\_text which makes it possible to assign every fetched tweet to exactly the follower who has sent this tweet.

*Naming convention for storing DataFrames in csv files*

For each different topic which is entered a new folder – called like the topic/querystring - will be created and within this folder a file with ‘*{topic}.csv’* will be created. In this file all fetched tweets for this topic are stored and therefor this is the container for the ‘Tweets’ DataFrame. Also, the ‘Users’ DataFrame is stored at the top level within this folder called like ‘*{topic}\_users.csv*’. If the user demands the followers of a user or the tweets and profiles of all followers a subfolder ‘followers’ is created in which all DataFrames are stored as csv with this naming convention: *{userid}\_followers.csv*. For the followers tweets the DataFrame will be stored also within this subfolder as {userid}\_*tweets\_followers.csv*.

*Markdown Files*

If the user demands to generate Markdown files those are all stored within a subfolder in the chosen topic folder called ‘*markdown*’.

## 3.4 Twitter Keys/Tokens

For loading Twitter Keys, Secrets and Token into the program the ‘os’ library will be utilized get those stored/exported as environment variables. Summed up, there are five environment variables to set to utilize the twitterclient: TWITTER\_ACCESS\_TOKEN\_SECRET, TWITTER\_ACCESS\_TOKEN, TWITTER\_CONSUMER\_SECRET, TWITTER\_CONSUMER\_KEY, TWITTER\_BEARER\_TOKEN.

The only required environment variable is the last one, the TWITTER\_BEARER\_TOKEN. For guaranteeing access all five environment variables should be set, but nonetheless it should work by merely setting this one required variable.

## 3.5 User Interface

For creating an easy to use and conveniently user-interface the library ‘simple-term-menu’ is leveraged. This library allows to easily create a main menu and several submenus where the user can interactively select available options by ‘up’ and ‘down’ keys or pressing the corresponding number.

Although an additional module is used for creating the user interface, it will completely run on the command line which simplifies input handling, e.g. the querystring/topic or choosing an tweet id to derive the polarity of. Additionally to mention is always the option of the user to generate markdown files for each DataFrame (see [3.3 Data Structures](#_3.3_Data_Structures)), which the user can open with an text editor of his choice to have a permanent file of desired request.

There is one big disadvantage of using this module which should not be concealed: The library ‘simple-term-menu’ is only supported by UNIX distributions and their command lines/shells. Therefor it’s not possible to run this program in Windows on PowerShell. But by installing Windows Subsystem Linux (WSL) it is also possible to run this program on the WSL.

# Class Diagram

Altogether there are four modules and each module contains a class.

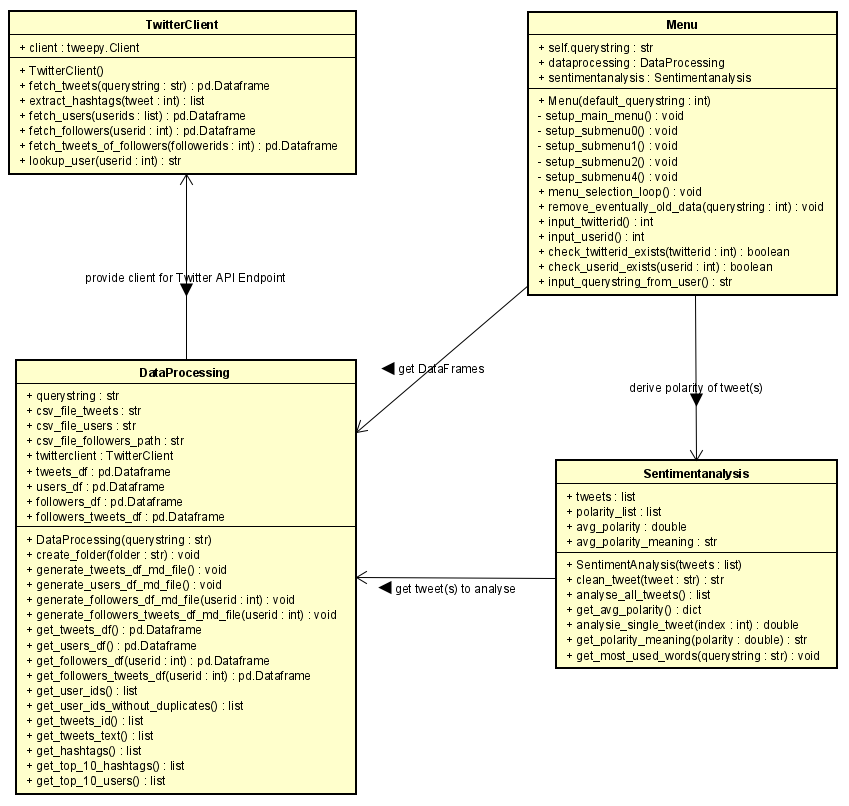
The ‘*TwitterClient*’ class provides a client authenticating to the Twitter API endpoint for fetching tweets, users, followers and tweets of followers. It is only responsible for fetching data from the Twitter API endpoint. If anything goes wrong by fetching data from Twitter it catches occurring errors and exceptions and displaying the user the cause of the issue.

The ‘*DataProcessing*’ class instantiates an instance of the TwitterClient class and uses it to fetch the data if it is not already available. This class is entirely responsible for loading the data as DataFrames into member variables either from csv files, if those are available, or to fetch it with the above mentioned instance of the TwitterClient. Furthermore, it handles the generation of markdown files and the calculation of the top 10 most hashtags/users as part of [Task 2](#Task2).

The ‘*Sentimentanalysis*’ class derives the sentiment, particularly the polarity, of each tweet and therefore fulfills [Task 1](#Task1). Furthermore it calculates the average polarity of each tweet, allows to analyse only a single tweet and additionally provides the meaning of the polarity as string. It also facilitates to get the most used words in the fetched dataset of tweets and creates a wordcloud with the mask of the official twitterlogo.

The ‘*Menu*’ class provides the interface between the program and the end user. It possesses an instance of the DataProcessing and Sentimentanalysis class. The first one is needed to get the DataFrames for displaying the data to the user, calculating top 10 hashtags/users and providing markdown files to the user whereas the latter is used to provide the user the possibility to analyze the desired tweet the user entered. This class is entirely responsible for the interaction with the user including creating the menu – main menu and all submenus – and retrieving input from the user. On the one hand for the selection of options in the menus is taken care of whereas on the other hand also text input is handled like getting a new querystring for changing the topic or entering a tweetid (or corresponding index in the DataFrame) to analyze or entering a userid (or corresponding index in the DataFrame) for getting the followers/tweets of followers of given user. Hence, this class provides and controls the entire output on the command line for the user.

The class diagram on the next page shows the relations between above four mentioned classes visually. It also shows the affiliated methods including their return types and member variables of each class.



# Listing of necessary functions for each Task

For better understanding I recommend to first read [4. Class Diagram](#_Class_Diagram) to get a clear view of the available classes/modules and their functionality. In following subchapters for each Task – Task 1 to Task 4 described in [2. The Project](#_The_Project_1) – the methods to fulfill this task are listed and described briefly. How the DataFrames are loaded from csv file or fetched with the twitterclient is documented in [6.1 Logic behind getting DataFrames](#_6.1_Logic_behind). For loading from csv file the DataProcessing methods *get\_{dataframe}\_df()* are needed and if not already stored as csv file the twitterclient methods *fetch\_{dataframe}()* are needed. For *{dataframe}* there are four options: *tweets*, *user*, *followers* and *followers\_tweets*.

**! Annotation:** As those functions are also commented in code with docstrings, the documentation of each method is documented with ‘*pydoc*’ and stored at ‘*docs/CodeDocumentation*’ whereas following subchapter only describe the relation between those functions and which methods are needed for which task.

## 5.1 Methods for Task 1

For [Task 1](#Task1), the derivement of the sentiment of each tweet, solely the class *Sentimentanalysis* is responsible. This class firstly gets all tweets as list in the constructor (in python \_\_init\_\_() function) and offers the method *analyse\_all\_tweets()* which analyses those tweets with the ‘*textblob’* library and returns their polarity also as list. This entirely fulfills Task 1 already as the User no can browse through the dataframe tweets including their polarity. Moreover, it is possible to analyse only a single tweet the user entered with the method *analyse\_single\_tweet()* doing exactly the same as the method for analysing all tweets but only for one tweet. Additionally the user can the average polarity with the *get\_avg\_polarity()* method which returns the average/mean polarity of the tweets in the acquired dataset. The last method *get\_most\_used\_words()*, kind of extra bonus for this Task, creates a wordcloudwith the officials twitterlogo mask of the 30 most used words and shows it the user.

## 5.2 Methods for Task 2

After the required DataFrame is loaded ([see 6.1 LogicBehindGettingDataFrames](#_6.1_Logic_behind)) [Task 2](#Task2) , getting the top 10 most used hashtags and 10 most users based on their number of tweets in the acquired dataset, can be fulfilled. This is done by leveraging the *DataProcessing* class and using the methods *get\_top\_10\_hashtags()* and *get\_top\_10\_users()* which are returning the 10 most used hashtags/users.

## 5.3 Methods for Task 3

For [Task 3](#Task3), getting the followers of a given user, only loading of the correct *followers* DataFrame is needed reading the csv file into a member variable of type pandas.DataFrame. Alternatively if the corresponding csv file does not exist yet, the twitterclient method *fetch\_followers(userid)* is called, where for a given userid the followers are fetched and returned as pandas DataFrame. This decision whether to load from csv file or fetching with the twitterclient is done within the method *get\_followers\_df()* of the *DataProcessing* class.

## 5.4 Methods for Task 4

For [Task 4](#Task4), obtaining the tweets and profiles of all followers of given user, only loading of the correct *followers\_tweets* DataFrame is need. This is either done by reading the corresponding csv file or fetching the data with the twitterclient method *fetch\_tweets\_of\_followers(followerids)* which returns a pandas DataFrame of tweets for given followerids of the given user. Afterwards it is stored into a member variable of type pandas.DataFrame of the *DataProcessing* class and subsequently printed out on the command line to the user.

# Program Sequence

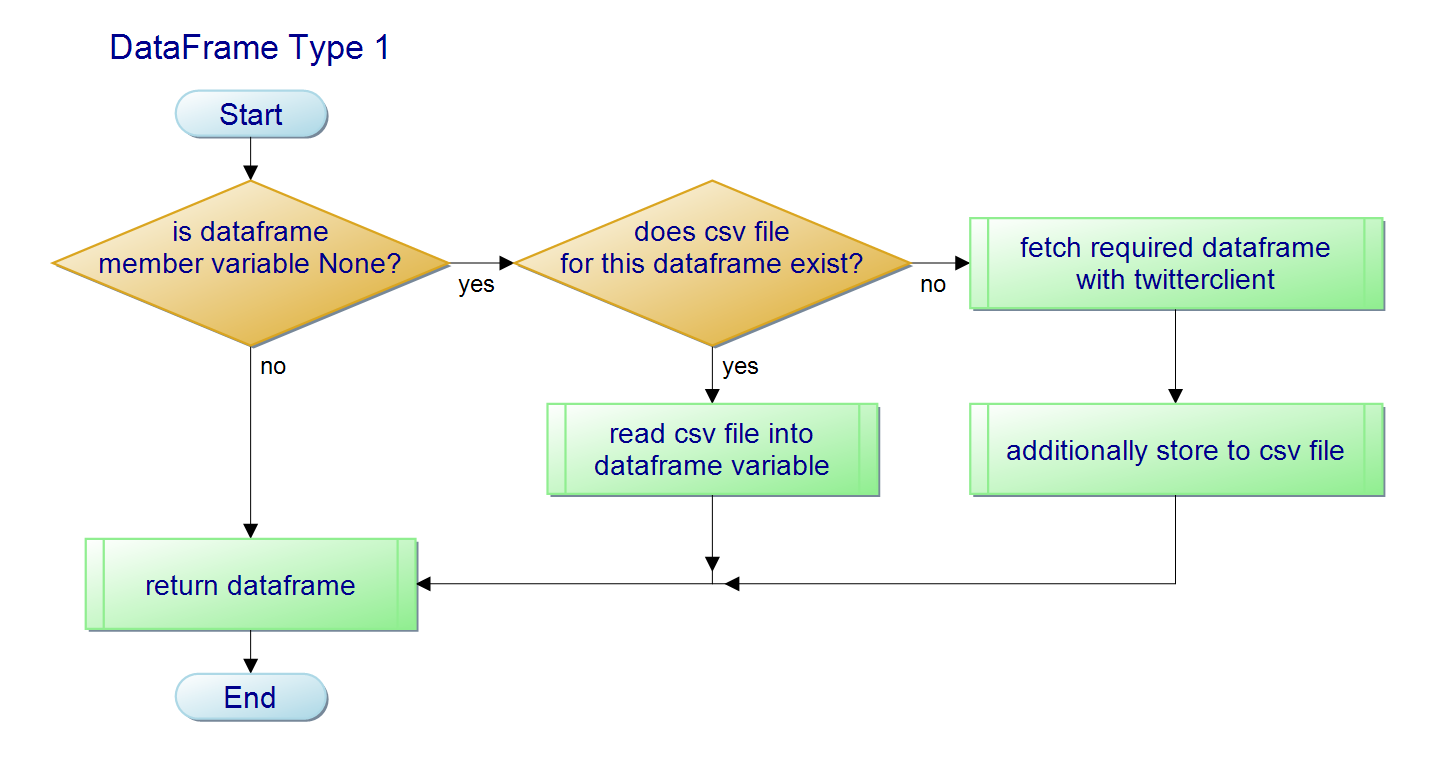
## 6.1 Logic behind getting DataFrames

As documented in [3.3 Data Structures](#_3.3_Data_Structures) there are four DataFrames. The *Tweets* and *Users* DataFrame are only loaded once either from an existing csv file or by fetching it with the twitterclient as they are permanent for one topic and are always the same. The other two DataFrames *Followers* and *Followers\_tweets* have to be reloaded for each different user id provided from the user. Therefore those are always loaded with the correct {userid}\_followers.csv respectively {userid}\_followers\_tweets.csv file if those already exists or corresponding requests with the twitterclient for given user ID are made.

Therefor those DataFrames can be divided into two types: the first [DataFrame Type 1](#_5.1.1_DataFrame_Type) considering the *Tweets* and *Users* DataFrame and the second DataFrame Type 2 considering the *Followers* and *Followers\_Tweets* DataFrame.

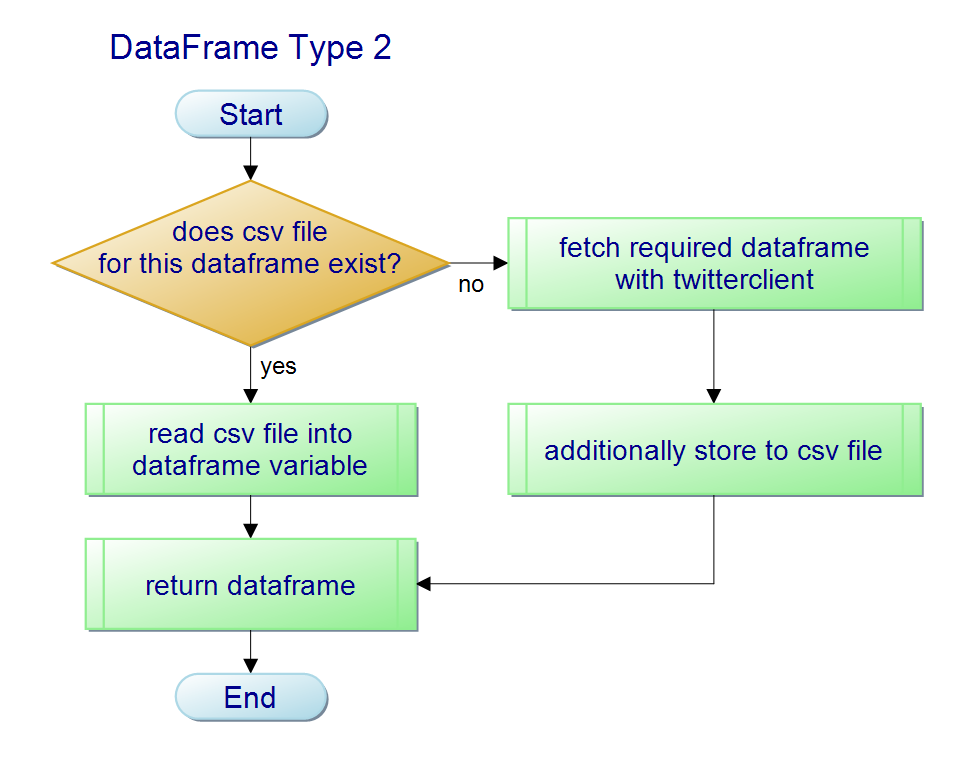
### 6.1.1 DataFrame Type 1

To this DataFrame Type 1 the *Tweets* and *Users* DataFrames are belonging to. Following flowchart explains the procedure getting this type of DataFrames: If the dataframe member variable of the DataProcessing class is already set the dataframe is directly returned. If not, then it will be checked if a csv file for corresponding topic and dataframe exists. If this csv file exists the csv file will be read with the *pandas read\_csv()* method and subsequently returned. If the csv file does not exist, the data for the required dataframe will be fetched with the twitterclient and additionally stored as csv file for the next time. Afterwards the dataframe is returned.



### 6.1.2 DataFrame Type 2

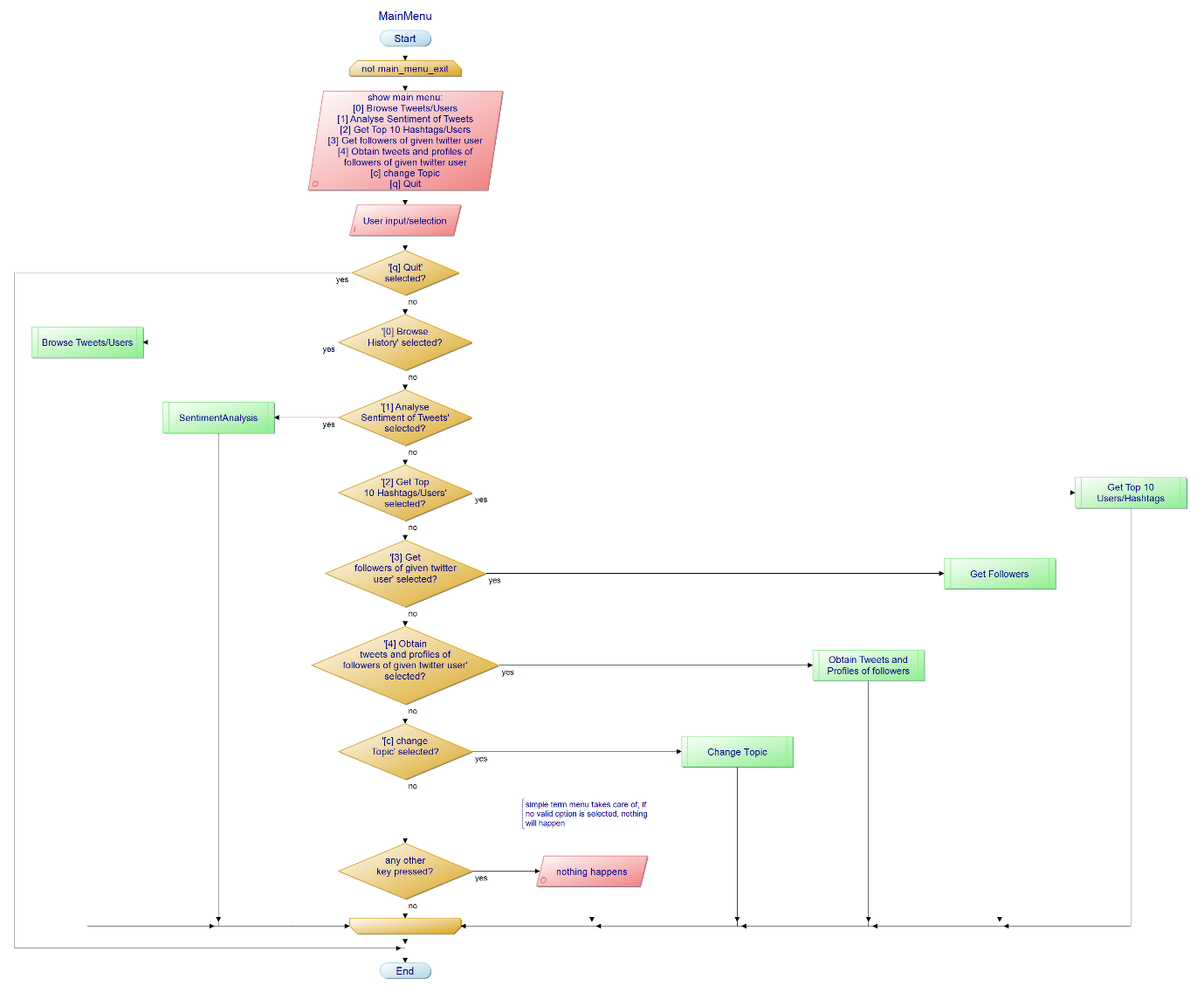
To this DataFrame Type 2 the *Followers* and *Followers\_Tweets* DataFrames are belonging to. Following flowchart explains the procedure getting this type of DataFrames: If a csv file for the corresponding topic and dataframe exists read it with the  *pandas read\_csv()* method and return the dataframe. If the csv file does not exist yet, the data for the required dataframe will be fetched with the twitterclient and additionally stored as csv file for the next time. Afterwards the dataframe is returned.



## 6.2 Menu

### 6.2.1 Main Menu

Following flowchart shows the main menu and its available options.



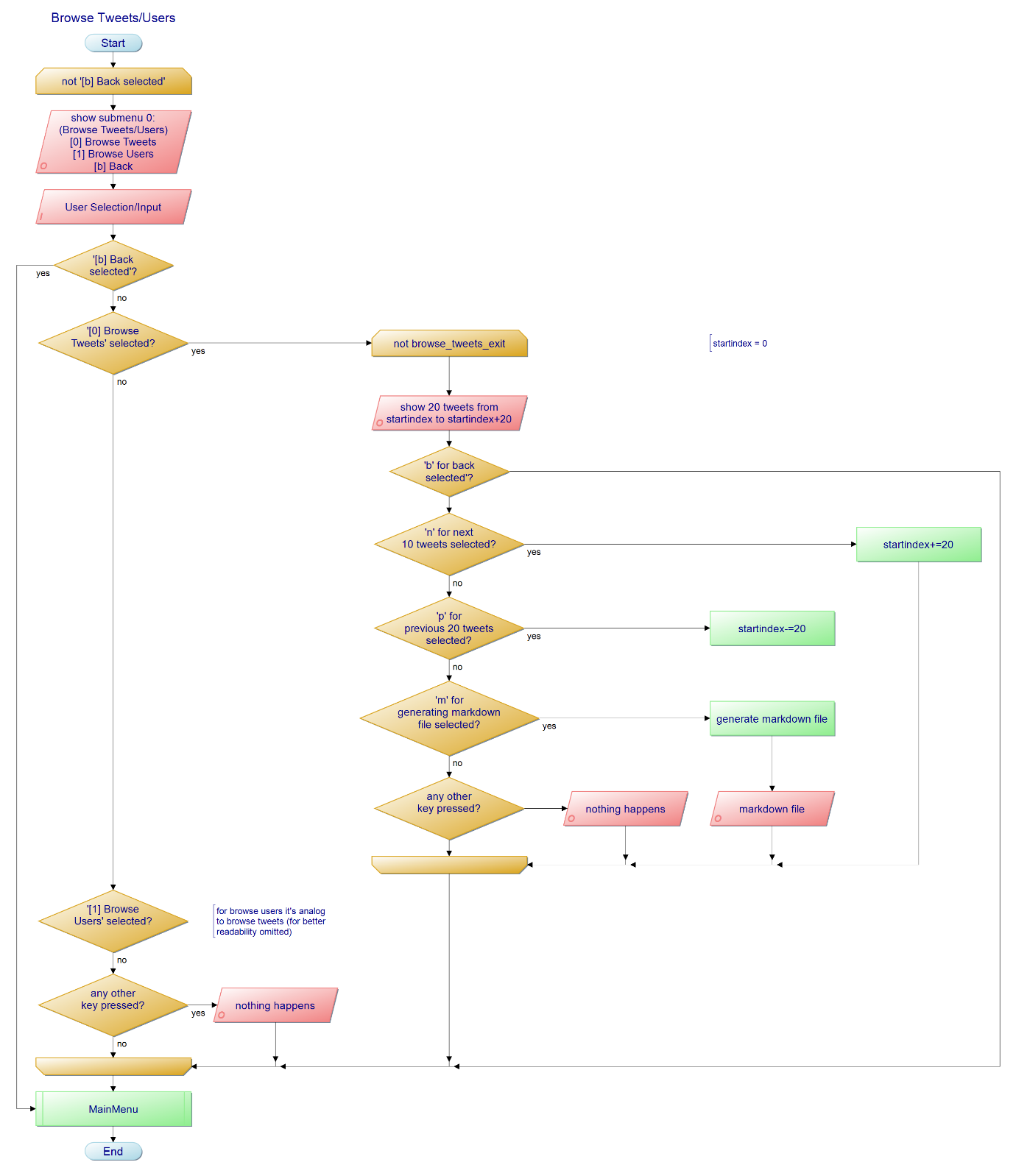
The user has the possibility to select one of seven choices:

If ‘[q] Quit’ is selected, the program quits. If ‘[0] Browse/Tweets’ is selected the user enters the Submenu 0 ([see 5.2.2](#_5.2.2_Browse_Tweets/Users)). If ‘[1] Analyse Sentiment of Tweets’ is selected the user enters the Submenu 1 ([see 5.2.3](#_5.2.3_Sentiment_Analysis)). If ‘[2] Get Top 10 Hashtags/Users’ is selected the user enters the Submenu 2 ([see 5.2.4](#_5.2.4_Get_Top)). If ‘[3] Get followers of given twitter user’ is selected the user enter the Submenu 3 ([see 5.2.5](#_5.2.5_Get_Followers)). If ‘[4] Obtain tweets and profiles of followers of given twitter user’ is selected the user enters Submenu 4 ([see 5.2.6](#_5.2.6_Obtain_tweets)). If ‘[c] change Topic’ is selected the user enters the change Topic submenu ([see 5.2.7](#_5.2.7_Change_Topic)) and can enter another topic, for which tweets will be fetched and subsequently processed and analyzed.

### 6.2.2 Browse Tweets/Users (Submenu 0)

In this submenu 0 – Browsing Tweets/Users – the user can either select to browse the acquired dataset of tweets or users. If the user has entered one of both options he can navigate through all tweets/users with keys ‘n’ for next and ‘p’ for previous 20 tweets/users. Additionally, the user has the possibility to get a markdown file of tweets/users generated and opened by pressing ‘m’. With the key ‘b’ the user gets back to the submenu 0 if he/she was browsing tweets or user.

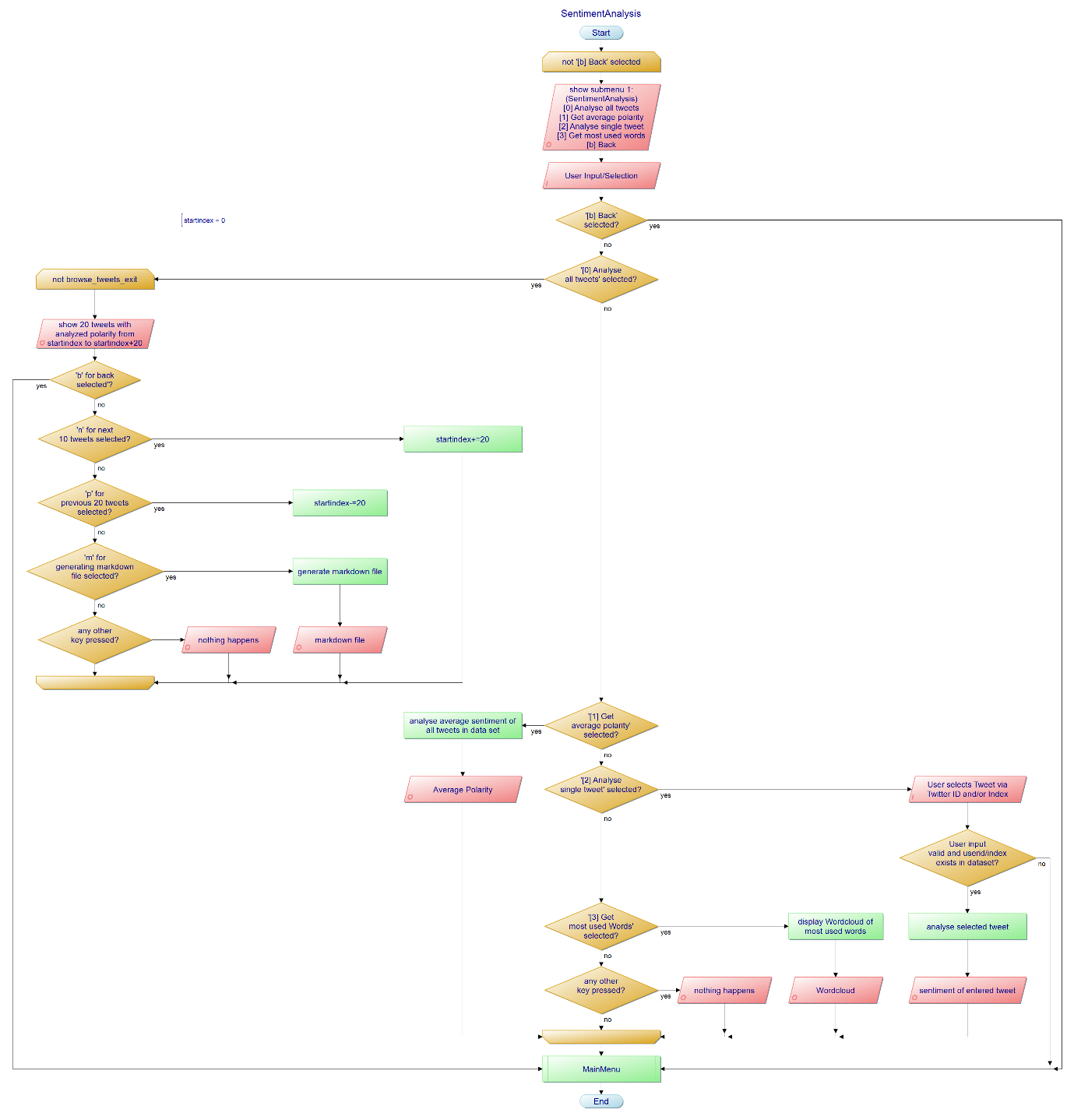
The ‘[b] Back’ option allows the user to go back to the main menu.



### 6.2.3 Sentiment Analysis (Submenu 1)

In this submenu 1 – Sentiment Analysis – the user can basically select between four options: ‘[0] Analyze all tweets’, ‘[1] Get average polarity’, ‘[2] Analyze single tweet’ and ‘[3] Get most used words’. First of these options analyzes all tweets in the dataset and let the user browse through the tweets and its analyzed polarity values. Second option displays the average polarity of all tweets in the acquired dataset. Third option allows the user to enter any Tweet ID or its corresponding index in the DataFrame and retrieve the polarity of desired tweet. The last of those three options generates a wordcloud with a mask of the official twitter logo including about 30 most used words in the tweets of the acquired dataset. Of course, this wordcloud will be subsequently shown to the user.

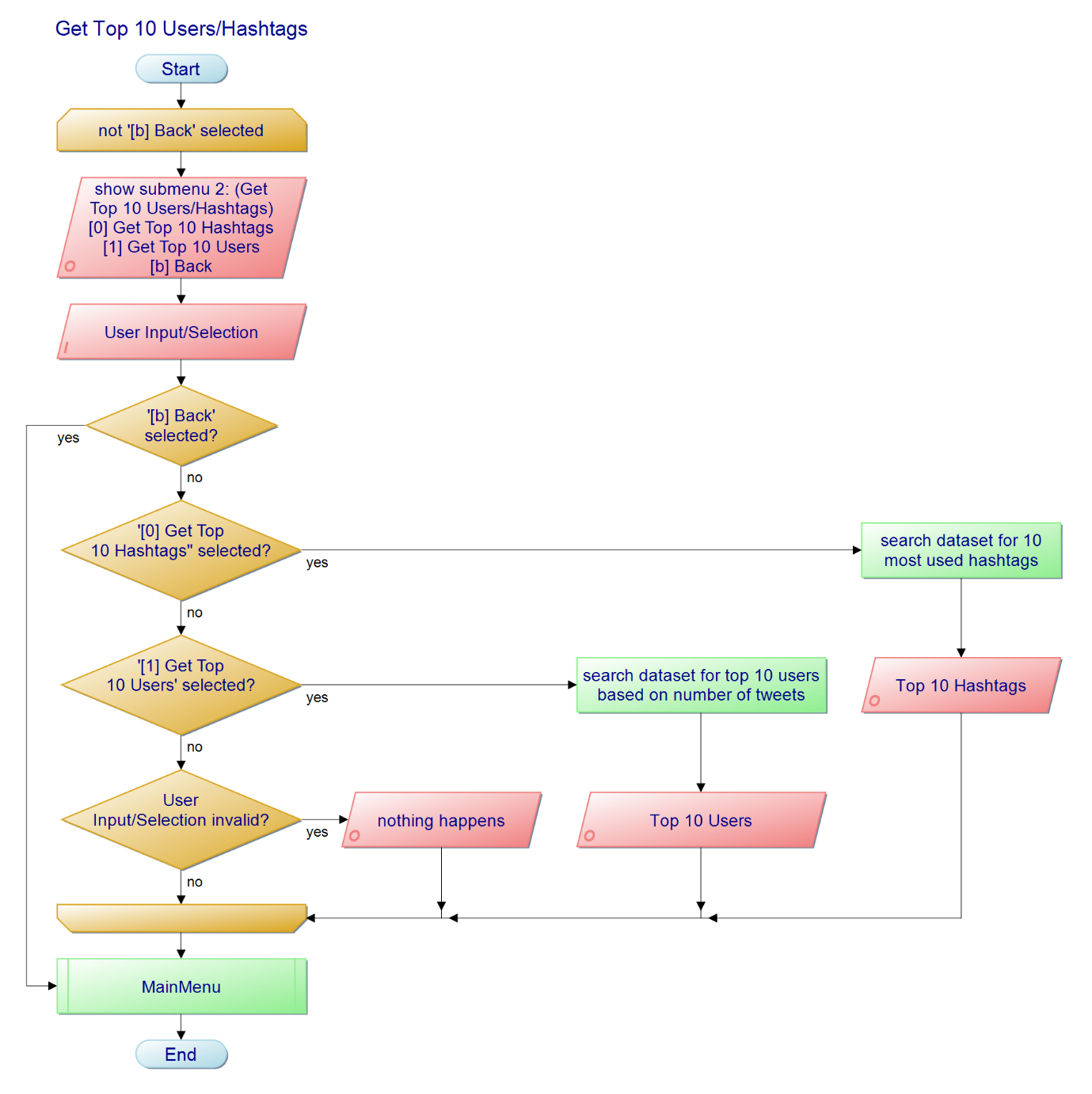
With the ‘[b] Back’ option the user gets back to the main menu.



### 6.2.4 Get Top 10 Users/Hashtags (Submenu 2)

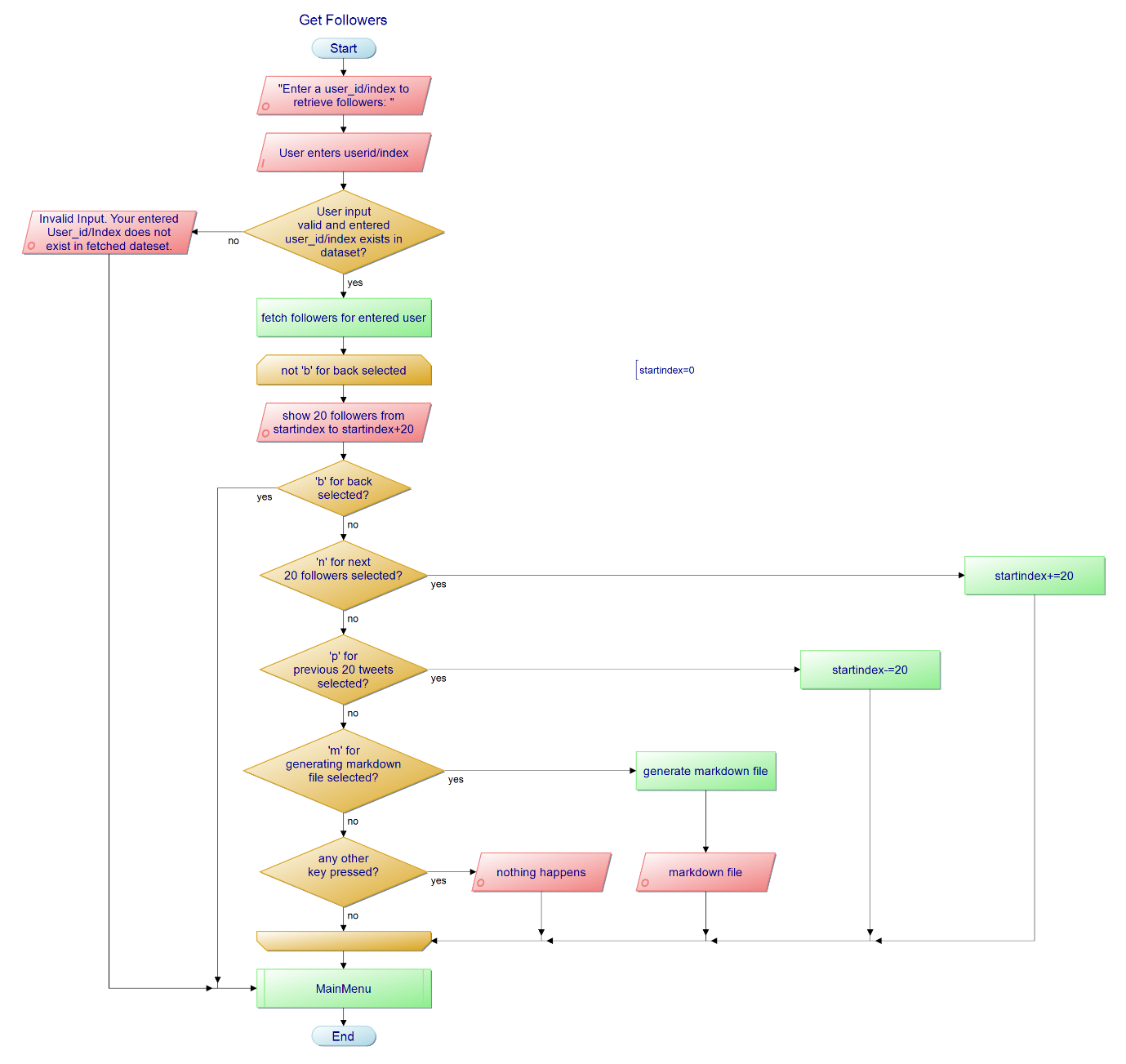
In the submenu 2 – Get Top 10 Users/Hashtags – the user can select between two options: ‘[0] Get Top 10 Hashtags’ and ‘[1] Get Top 10 Users’. The first choice prints out the username (+ user ID) and the number of tweets of the top 10 users based on the greatest number of tweets in acquired dataset. The latter choice prints out top 10 most used hashtags of the acquired dataset.

With the ‘[b] Back’ option the user gets back to the main menu.

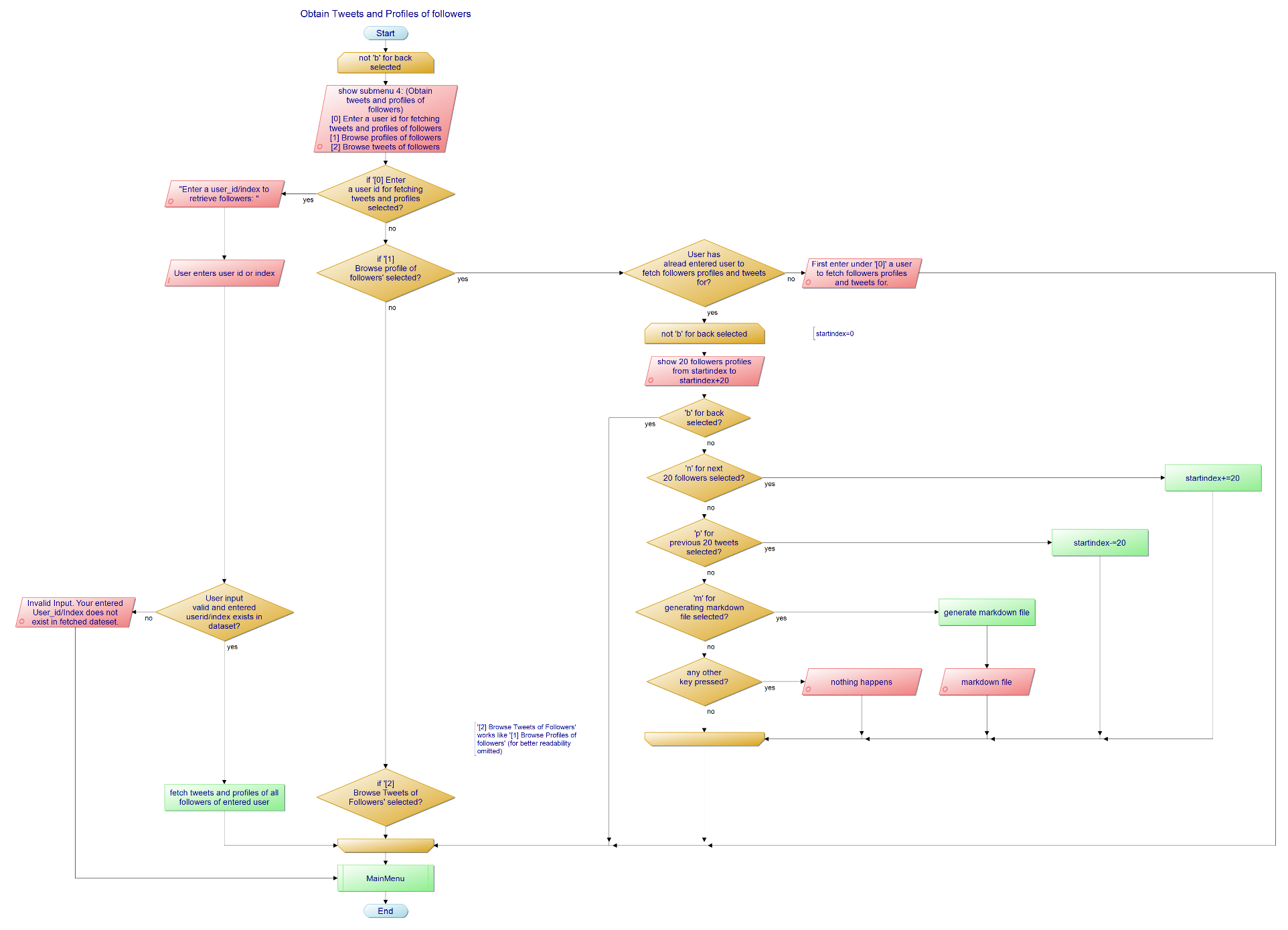


### 6.2.5 Get Followers (Submenu 3)

In the submenu 3 – Get Followers – the user can get followers of the entered user. The user can either enter the user ID or the index of the user in the dataframe. If the input is not valid the program will respond with an error message. Providing that the entered user is valid and available in the acquired dataset he/she will retrieve user ID, username and name of all followers. But to not overload the entire output the user can paginate through all followers in steps of 20 followers with the ‘n’/’p’ for next/previous 20 followers. With the ‘m’ option the user can demand a markdown file of all followers and look through it in a text edit of his/her choice.

With the ‘b’ option the user gets back to the main menu.

### 6.2.6 Obtain tweets and profiles of followers of given twitter user (Submenu 4)

In the submenu 4 - Obtain tweets and profiles of followers of given twitter user – the user can enter under ‘[0] Enter a user ID for fetching tweets and profiles of followers’ a user ID or the index of the user in the dataframe and if valid, the program will fetch followers including their profiles and about 20 tweets of each follower. (This amount of tweets per follower can be changed.)

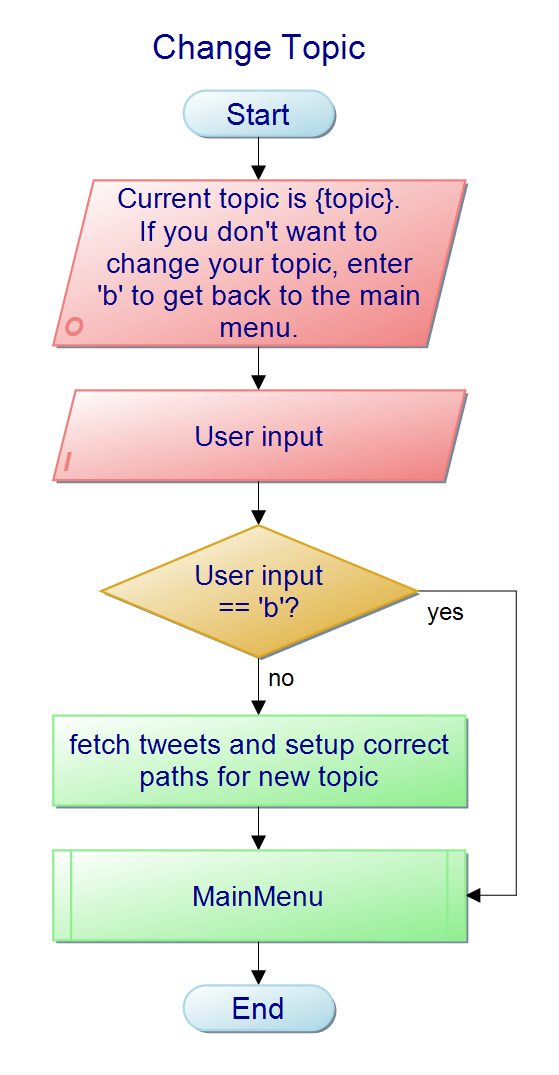
Under ‘[1] Browse profiles of followers’ the user can paginate in steps of 20 followers through the profiles of all followers with ‘n’/’p’ for next/ previous page. Under ‘[2] Browse tweets of followers’ the user can do basically the same but for the tweets of all followers. Presumably for each follower about 20-50 tweets are fetched. Moreover the user can get a markdown file for the last two options by pressing ‘m’.-By pressing the key ‘b’ he/she will get back to the submenu 4.

If the user is in the submenu 4, he/she can get back to the main menu by selecting the option ‘[b] Back’.

### 6.2.7 Change Topic

The default topic is ‘computer’, also shown in the main menu, and for this topic tweets are already prefetched and stored as csv files under directory ‘*fetched/computer*’. Moreover, the instantiations of classes for dataprocessing and sentimentanalysis are done in advance.

But to facilitate it for the user to also use this program for other topics respectively topics of interest of the user it is possible to change the topic. The user can change the topic to (almost) everything but has to accept to retrieve nothing if the topic entered is chosen unfavorable (like only % symbols).

If the user presses enter, he/she will immediately get back to the main menu without choosing a new topic. But if a topic is entered, tweets for the new topic are fetched including also instantiating a dataprocessing and sentimentanalysis object. Additionally, the paths of the new topic are configured within the DataProcessing instance to access desired csv file with given naming convention correctly ([see 3.3](#_3.3_Data_Structures)).

# Results

The menu and all corresponding submenus are already implemented and in the screenshot below you can see the main menu.

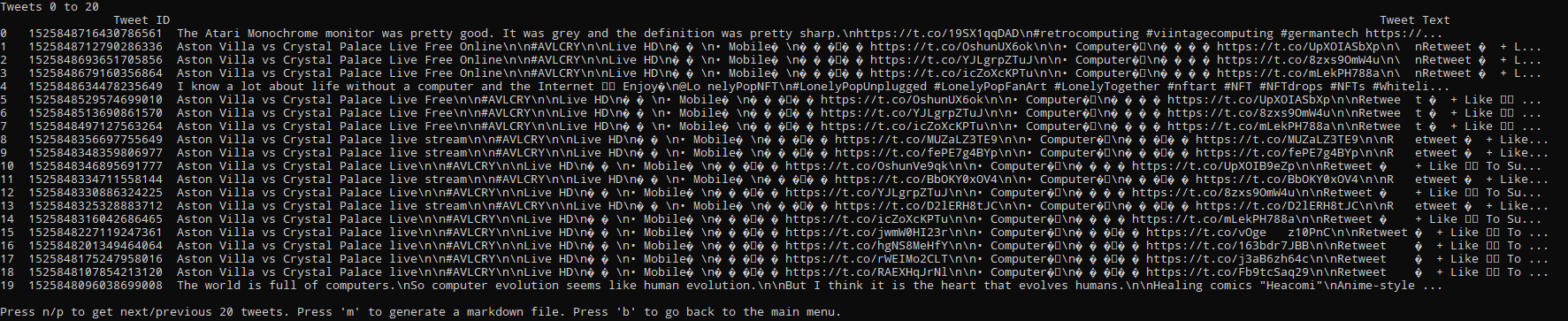
Ein Bild, das Text enthält.

Automatisch generierte Beschreibung  
Screenshot 1: Main Menu

The option ‘[0] Browse Tweets/Users’ is completely implemented except browsing users, as the User Dataframe is not fetched and stored yet. See screenshots below.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung  
Screenshot 2: Submenu 00 – Browse Tweets/Users

  
Screenshot 3: Submenu 00 – Browse Tweets - Example

The options ‘[c] change Topic’ and ‘[q] Quit’ are also completely implemented. The latter just simply quits the program and the first allows the user to enter another topic and enable all available options for the entered topic. This includes dynamically setting up the twitterclient, dataprocessing and sentimentanalysis for the new topic.

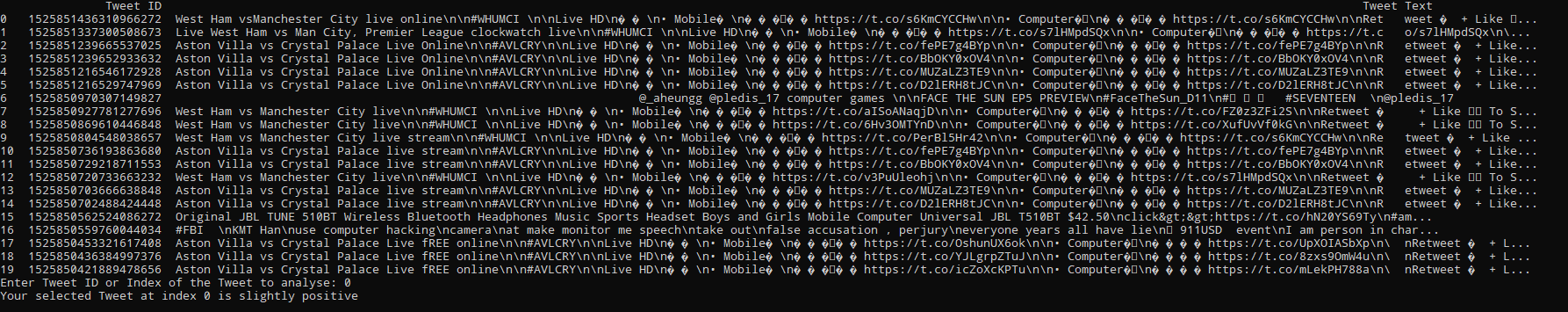
[**Task 1**](#Task1)**:** *Derive the sentiment of each tweet.*

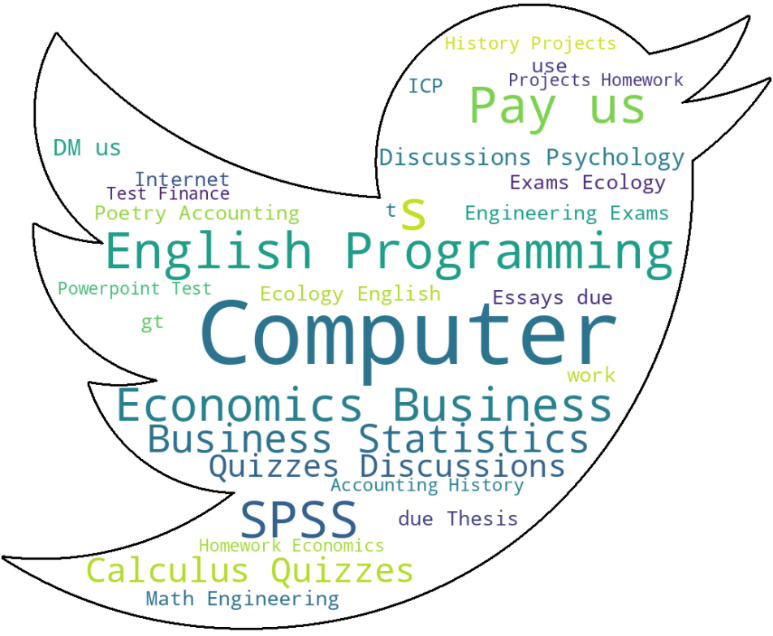
The option ‘[1] Analyse Sentiment of Tweets’ is entirely implemented and presumably only some refactoring is required. Thus, it is possible for the user to get the average polarity of all tweets, to enter a tweet\_id (alternatively the index of the corresponding tweet) to only analyze the entered tweet or to show the user the 30 (maybe the number will change) most used words in all tweets in a wordcloud with the mask of the twitterlogo.

Following three screenshots will prove above stated progress. Screenshot 4 shows the Submenu 01 for Sentimentanalysis. Screenshot 5 shows an example foro choosing ‘[1] Analyse single tweet’ and the last screenshot (screenshot 6) shows the output of ‘[2] Get most used words’.

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungScreenshot 4: Submenu 01 - Sentimentanalysis

Screenshot 5: Submenu 01 - Sentimentanalysis – example for analysing single tweet

Screenshot 6: Submenu 01 – Sentimentanalysis – example for getting most used words as a wordcloud in form of the twitterlogo 

[**Task 2**](#Task2)**:** *Get Top 10 Hashtags/Users based on their number.*

The option ‘[2] Get Top 10 Users/Hashtags’ for corresponding Task 2 is almost completely implemented. Only for the 10 Users there is currently the user ID shown as response which I will refactor to lookup the username of corresponding user ID.

Screenshot 7 shows the submenu 2 for getting top 10 Hashtags respectively Users. Screenshot 8 and Screenshot 9 show an example for both options.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung  
Screenshot 7: Submenu 02 – Get Top 10 Hashtags/Users

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung  
Screenshot 8: Submenu 02 – Get Top 10 Hashtags/Users – example for Top 10 Hashtags

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung  
Screenshot 9: Submenu 02 – Get Top 10 Hashtags/Users – example for Top 10 Users

[**Task 3**](#Task3)**:** *Get the followers of given twitter user.*

The implementation of this task is currently in progress. The approach is to let the user enter a user\_id or its corresponding index in the dataframe and fetch all followers. All these followers are displayed, and the user can paginate with n/p to the next/previous 20 followers. The followers will presumably be stored in a dataframe and/or additionally in a csv file.

[**Task 4**](#Task4)**:** *Given a twitter user, obtain the tweets and profiles of all followers of the user and show it.*

This task is not implemented yet. Only approaches how to solve this task were gathered and partially tried out in some requests.

# Source Code / Instructions

The source code of the project is zipped within the submission of this task three. Anyway this project is also stored in a GitHub repository: <https://github.com/danielgruber99/TwitterDataAnalysis>.  
 ***This project only works on Linux Systems due the use of the simple-term-menu library! Please use either a Linux VM (of any distribution) or setup the Windows Subsystem Linux (WSL) on Windows.***

**Instructions:***Preliminary remark*: Either unzip the source code submitted in this task or clone the GitHub Repository (see link above). All Instructions can be also found in the Readme file along with the requirements in requirements.txt.

1. If not yet done, create a Twitter developer account and retrieve your access and consumer token/keys and secrets. Moreover, a bearer token is created by creating a Twitter developer App. Store those credentials as environment variables by running:   
 ‘*export <NameOfVariable>= <ValueofVariable>*’ for following five variables:

* TWITTER\_ACCESS\_TOKEN\_SECRET = <YourAccessTokenSecret>
* TWITTER\_ACCESS\_TOKEN = <YourAccessToken>
* TWITTER\_CONSUMER\_SECRET = <YourConsumerSecret>
* TWITTER\_CONSUMER\_KEY = <YourConsumerKey>
* TWITTER\_BEARER\_TOKEN = <YourBearerToken>

2. Setup the virtual environment in the source code by running: ‘*. .venv/bin/activate*’. This virtual environment has all required modules/libraries installed and therefore no further modules should be needed. If any problems occur try to install modules manually with ‘*pip install -r requirements.txt*’.

3. Start the program by running ‘*python3 main.py*’. Now you are ready to utilize every function of this program.

## 4.4 Outlook

Currently only possible to fetch maximum 100 tweets (implement by looping over … to fetch more than 100 tweets (as max\_results is allowd only to 100))

Refactor lookup\_user method to to make api calls if user df already exists (or csv file).

Keyboard library for better bedienbarkeit