**Final C Project Report**

**Tweet Analysis on Elon Musk Dataset**

**(Final Draft)**

**CPEN 220**

**Programming for Engineering Solutions**

*Malek Wahidi*

The final draft of this project features a fully functional C program, MyTweets, which can perform all sorts of analyses on Elon Musk tweets from a Kaggle dataset. The project is neatly organized into multiple folders according to standard conventions for C projects. The ‘src’ folder contains the main program, along with a couple of c files that contain relevant functions for loading data and performing tweet analysis. The ‘headers’ folder contains a header file that includes function and struct declarations, and macro definitions, to be shared with the source files. The ‘resources’ folder contains program resources such as the Kaggle dataset for Elon Musk tweets and the VADER dictionary used for tweet sentiment analysis.

The original ‘cleandata.csv’ file presented many complications in reading the dataset correctly. Multiple exceptions created bugs such as (\n) or (,) characters in the middle of tweets. As a simple solution to those problems, I decided to remove the raw\_tweets column since all the analysis will be performed on the cleaned tweets for maximized efficiency. In addition, the clean\_tweets column was moved to the left to become the first column in the table, in order to simplify the task of detecting quoted tweets at the beginning of each new line.

The user can interact with MyTweets through a command-line interface (CLI) which accepts input arguments for selecting the required option and providing input arguments when needed. This interface can be used to run a range of operations ranging from statistical analysis, to search features, list sorting, and even sentiment analysis when possible. All these functions could be found in the ‘functions.c’ source file and are listed and described in the help text made available by using the –h or --help arguments.

The exact functions featured in MyTweets include printing the entire organized dataset of tweets or printing a single tweet. For statistical analysis, the program can find the least and most liked (or retweeted) tweets, in addition to calculating the average likes or tweets for the entire dataset. For search functionality, MyTweets can search the text of each tweet throughout the entire dataset to find a matching keyword given by the user. Alternatively, it can search for tweets by a given date, month, or even a specific time. Finally, a ‘save’ functionality was added to MyTweets which allows the user to save all printed tweets in the functions used after ‘–save’ to a local text file for later analysis. Due to the ‘append’ mode used here, the save feature allows the user to save multiple outputs from many different functions to the same text file.

Finally, to perform tweet analysis, MyTweets loads words and their corresponding sentimental polarities (positive or negative values) from a VADER dictionary text file acquired from GitHub. The text file was converted to CSV format using Excel 2016 in order to allow for efficient reading and saving of the data into two separate arrays. MyTweets first normalizes the polarity values (originally between -4 and 4) to a range between -1 and 1 for more intuitive scores. When the user provides a given tweet ID on the CLI, MyTweets finds the corresponding tweet, extracts the text content of that tweet, and then searches through the loaded dictionary for each word found in that tweet. A function was also implemented to filter out punctuation when found alongside a string to allow for a proper search. For each word from the tweet that is found in the 7,518-word dictionary, the respective sentimental polarity is acquired. The polarities for all detected words are averaged out and a final average polarity score is calculated. An overall positive polarity score indicates a generally positive tweet, while a negative one reveals the opposite. A polarity score close to 0 indicates an emotionally neutral tweet.