```
import sys
import json
import re
from collections import defaultdict

## Import the US States file

states = {
        'AK': 'Alaska',
        'AL': 'Alabama',
        'AR': 'Arkansas',
        'AS': 'American Samoa',
        'AZ': 'Arizona',
        'CA': 'California',
        'CO': 'Colorado',
        'CT': 'Connecticut',
        'DC': 'District of Columbia',
        'DE': 'Delaware',
        'FL': 'Florida',
        'GA': 'Georgia',
        'GU': 'Guam',
        'HI': 'Hawaii',
        'IA': 'Iowa',
        'ID': 'Idaho',
        'IL': 'Illinois',
        'IN': 'Indiana',
        'KS': 'Kansas',
        'KY': 'Kentucky',
        'LA': 'Louisiana',
        'MA': 'Massachusetts',
        'MD': 'Maryland',
        'ME': 'Maine',
        'MI': 'Michigan',
        'MN': 'Minnesota',
        'MO': 'Missouri',
        'MP': 'Northern Mariana Islands',
        'MS': 'Mississippi',
        'MT': 'Montana',
        'NA': 'National',
        'NC': 'North Carolina',
        'ND': 'North Dakota',
        'NE': 'Nebraska',
        'NH': 'New Hampshire',
        'NJ': 'New Jersey',
        'NM': 'New Mexico',
        'NV': 'Nevada',
        'NY': 'New York',
        'OH': 'Ohio',
        'OK': 'Oklahoma',
        'OR': 'Oregon',
        'PA': 'Pennsylvania',
        'PR': 'Puerto Rico',
        'RI': 'Rhode Island',
        'SC': 'South Carolina',
        'SD': 'South Dakota',
        'TN': 'Tennessee',
```

```
        'TX': 'Texas',
        'UT': 'Utah',
        'VA': 'Virginia',
        'VI': 'Virgin Islands',
        'VT': 'Vermont',
        'WA': 'Washington',
        'WI': 'Wisconsin',
        'WV': 'West Virginia',
        'WY': 'Wyoming'
}

def tweet_score(tweet,scores):
    ## given a tweet text and sentiment score file, returns tweet sentiment
    ## score word in tweet is in scores or 0 if not

    return sum(scores.get(word,0) for word in tweet[0].split())

def tweet_parser(tweet):

    ## Extracts state, text from a tweet input as a python object

    ## Define search strings

    srch_country=r'(?<=country_code":)"(.*?)"'
    srch_state_loc=r'(?<=location":)"(.*?)"'
    srch_state_fname=r'(?<=full_name":)"(.*?)"'
    srch_tweet=r'(?<=text":)"(.*?),"'

    ## Define output variables state

    state=None

    ## Return state , text for a tweet

    try:
        ## State from location field, if it exists

        for word in re.findall(srch_state_loc,tweet)[0].split():
            for key in states:
                    if word ==states[key] or word ==key:
                            state= key
                            text=re.findall(srch_tweet,tweet)
                            return state,text

        ## State from place and its sub-fields, if they
        ## exist and location is blank.
        if not state:
            if re.findall(srch_country,tweet):
                if re.findall(srch_country,tweet)[0]=='US':
                    for word in re.findall(srch_state_fname,tweet)[0].split(','):
                        for key in states:
                            if word ==key or word ==states[key]:
                                state=key
                                text=re.findall(srch_tweet,tweet)
                                return state,text
    except (KeyError, TypeError, IndexError):
```

```
        return None

def main():

    ##Open input text files

    sent_file = open(sys.argv[1]) #file containing sentiment words and values
    tweet_file = open(sys.argv[2]) # tweet data streamed from twitter site


    ## Populate scores with sentiment values from sent_file

    sent_file.seek(0)
    scores={line.split('\t')[0]:int(line.split('\t')[1])
                for line in sent_file}


    ## Process input tweet_file to find happiest state

    tweet_file.seek(0)

    ## Initiate dictionaries to store state parameters

    twt_cnt_state=defaultdict(float)
    twt_scr_state=defaultdict(float)
    happy_scr_state=defaultdict(float)

    ## Returns (prints) the happiest state code or message

    ## Parse Tweet data to get tweet text and state

    tweets_states=(tweet_parser(twt) for twt in tweet_file
                    if tweet_parser(twt))
    score_states=((state,tweet_score(text,scores)) for state, text
                  in tweets_states)

    ## Roll up scores for each state found and calculate average

    for state,score in score_states:
        twt_cnt_state[state]+=1
        twt_scr_state[state]+=score
        happy_scr_state[state]=twt_scr_state[state]/twt_cnt_state[state]

    ## Print final happiest state, if error print default

    try:
        print max(sorted(happy_scr_state.keys()),key=happy_scr_state.get)
    except ValueError:
        print ('NY')


if __name__ == '__main__':
    main()
```