In [39]:
```
# Date of Birth in DD-MM-YYYY format is 07-07-2000
```

In [17]:
```
import PyPDF2

# Function to extract pages and save them as a text file
def extract_pages_to_text(pdf_file, start_page, num_pages, output_file)
    with open(pdf_file, 'rb') as file:
        reader = PyPDF2.PdfReader(file)
        extracted_text = ""

        # Loop to extract the specified pages
        for i in range(start_page - 1, start_page - 1 + num_pages):
            extracted_text += reader.pages[i].extract_text()

        # Write the extracted text to the output file
        with open(output_file, 'w') as txt_file:
            txt_file.write(extracted_text)

# File path for the Harry Potter PDF
pdf_path = '/Users/reddysaireddyduggireddy/Downloads/Harry_Potter_(www.

# Assuming Book 7 starts on page 1500 (adjust based on your check)
book_7_start_page = 5946

# For the birth date (07), extract 10 pages starting from page 7 of Boo
extract_pages_to_text(pdf_path, book_7_start_page + 7, 10, 'file1.txt')

# For the birth year (2000), extract 10 pages starting from page 100 of
extract_pages_to_text(pdf_path, book_7_start_page + 100, 10, 'file2.txt

print("Pages successfully extracted to file1.txt and file2.txt")
```

```
Pages successfully extracted to file1.txt and file2.txt
```

# Question-1- count occurrences of each word

In [18]:
```
import re
from collections import defaultdict

# Read file
with open('file1.txt', 'r') as file:

# Preprocessing: Convert text to lowercase and remove punctuation, numb
def preprocess(text_file):
    text_file = text_file.lower()
    cleaned_text = re.sub(r'[^a-zA-Z\s]', '', text_file)
    return cleaned_text

processed_text = preprocess(text_file)

# Mapper function
def mapper(text):
    words = text.split()
    word_counts = defaultdict(int)
    for word in words:
        word_counts[word] += 1
    return word_counts

# Reducer function
def reducer(mapped_data):
    final_counts = defaultdict(int)
    for word, count in mapped_data.items():
        final_counts[word] += count
    return final_counts

# MapReduce steps
map_data = mapper(processed_text)
reduce_data = reducer(map_data)

sorted_dict = dict(sorted(reduce_data.items(), key=lambda x: x[0]))

print(sorted_dict)
```

{'a': 27, 'about': 3, 'above': 2, 'absently': 1, 'according': 1, 'acro
ss': 2, 'act': 1, 'addressing': 1, 'advantage': 1, 'afraid': 1, 'afte
r': 2, 'again': 4, 'ahead': 1, 'all': 5, 'along': 1, 'also': 1, 'an':
7, 'and': 35, 'angrily': 1, 'announced': 1, 'any': 3, 'anymore': 1, 'a
pparates': 1, 'appeared': 2, 'apprehensively': 1, 'approval': 1, 'ar
e': 2, 'arms': 1, 'around': 4, 'as': 13, 'at': 17, 'attempt': 1, 'atte
nd': 1, 'averted': 1, 'away': 1, 'axley': 5, 'back': 4, 'barely': 1,
'be': 12, 'bearing': 1, 'been': 7, 'before': 6, 'behind': 1, 'bellatri
x': 6, 'below': 1, 'beneath': 2, 'beside': 1, 'best': 1, 'better': 3,
'blamed': 1, 'blonde': 1, 'blotchy': 1, 'body': 4, 'borrow': 2, 'boy':
3, 'break': 1, 'briefly': 1, 'bring': 1, 'but': 11, 'by': 7, 'called':
1, 'came': 1, 'can': 2, 'cannot': 1, 'careless': 1, 'chair': 2, 'chanc
e': 2, 'change': 1, 'clapped': 1, 'climb': 1, 'closed': 1, 'closely':
1, 'closeness': 1, 'color': 1, 'company': 1, 'compared': 2, 'concerne
d': 1, 'confused': 1, 'considered': 1, 'constricted': 1, 'contact': 2,
'continued': 2, 'control': 1, 'controlled': 1, 'converted': 1, 'core':
1, 'could': 3, 'course': 2, 'creature': 1, 'cried': 1, 'cruel': 1, 'cr
y': 1, 'curious': 1, 'curse': 1, 'dark': 2, 'deal': 1, 'death': 1, 'de
athly': 6, 'delight': 1, 'demeanor': 1, 'demonstrate': 1, 'departmen
t': 2, 'departments': 1, 'desire': 2, 'destination': 1, 'determined':
1, 'did': 3, 'dif': 1, 'discover': 1, 'discovered': 1, 'disliking': 1,
'displayed': 1, 'displeases': 1, 'disturbance': 1, 'do': 4, 'dolohov':
1, 'don': 1, 'done': 1, 'down': 5, 'downward': 1, 'draco': 1, 'drago
n': 2, 'drawnout': 1, 'drew': 1, 'due': 1, 'e': 10, 'each': 1, 'easie
r': 1, 'easy': 1, 'eaters': 1, 'ef': 1, 'either': 1, 'ell': 1, 'elm':
1, 'emer': 1, 'emotion': 1, 'enchantments': 1, 'endlessly': 1, 'enforc
ement': 1, 'enough': 2, 'errors': 1, 'eruption': 1, 'es': 2, 'eschewin
g': 1, 'even': 2, 'event': 1, 'every': 1, 'everything': 1, 'evidentl
y': 1, 'examining': 1, 'exchange': 2, 'existence': 1, 'expected': 1,
'expression': 1, 'eye': 1, 'eyes': 10, 'f': 1, 'face': 3, 'faces': 2,
'failed': 1, 'fallen': 2, 'family': 3, 'far': 1, 'feet': 1, 'few': 1,
'ficial': 1, 'ficulty': 1, 'fingers': 2, 'firelight': 2, 'first': 1,
'fists': 1, 'floo': 1, 'flooded': 1, 'floor': 1, 'flushed': 1, 'follow
ers': 1, 'for': 11, 'form': 1, 'fort': 1, 'forward': 1, 'fraction': 1,
'friend': 1, 'from': 9, 'front': 1, 'g': 6, 'gasped': 1, 'gazing': 1,
'ge': 1, 'ged': 1, 'gesture': 1, 'give': 2, 'given': 2, 'glance': 1,
'glanced': 2, 'gleam': 1, 'gleeful': 1, 'glinting': 1, 'go': 1, 'goin
g': 1, 'good': 2, 'great': 3, 'grew': 1, 'had': 6, 'hair': 2, 'halfwa
y': 2, 'hallows': 6, 'hand': 3, 'hanging': 1, 'happiness': 1, 'happy':
2, 'harry': 9, 'has': 6, 'have': 17, 'he': 15, 'head': 2, 'heads': 2,
'hear': 2, 'heard': 1, 'heartstring': 1, 'heavily': 1, 'heavy': 1, 'he
ld': 2, 'her': 14, 'here': 1, 'hide': 1, 'higher': 3, 'highranking':
1, 'him': 6, 'himself': 2, 'his': 27, 'hiss': 1, 'hissed': 1, 'hissin
g': 1, 'hoarse': 1, 'home': 2, 'honor': 1, 'house': 1, 'however': 1,
'huge': 1, 'humiliation': 1, 'i': 26, 'if': 2, 'ill': 1, 'im': 1, 'imm
ediately': 1, 'impassive': 1, 'imperius': 1, 'impressed': 1, 'in': 15,
'inert': 1, 'instance': 1, 'into': 1, 'involuntary': 1, 'is': 16, 'iss
ue': 1, 'it': 15, 'its': 3, 'jeering': 1, 'jk': 6, 'jubilant': 1, 'jus
t': 1, 'keeping': 1, 'kill': 2, 'know': 4, 'knows': 1, 'laid': 1, 'la
r': 1, 'late': 1, 'laughter': 1, 'law': 1, 'leaned': 2, 'leaving': 1,
'left': 1, 'lengths': 1, 'less': 1, 'let': 1, 'liberty': 1, 'lidded':
1, 'lies': 1, 'life': 1, 'lip': 1, 'lips': 1, 'little': 2, 'lives': 1,
'long': 5, 'longing': 1, 'look': 1, 'looked': 4, 'looking': 2, 'look
s': 2, 'lord': 13, 'lot': 1, 'louder': 1, 'low': 1, 'lucius': 11, 'luc
k': 1, 'lupin': 1, 'made': 2, 'magical': 2, 'make': 1, 'malfoy': 8, 'm
alfoys': 2, 'maliciously': 1, 'man': 4, 'many': 5, 'married': 2, 'me':
1, 'mean': 1, 'means': 1, 'mere': 1, 'might': 3, 'mine': 1, 'ministe

r': 2, 'ministry': 6, 'mirth': 1, 'misery': 1, 'missed': 1, 'mistake
s': 1, 'mistrust': 1, 'mmy': 1, 'more': 2, 'mouth': 2, 'move': 1, 'mov
ement': 1, 'moving': 1, 'must': 4, 'my': 19, 'narcissa': 3, 'neck': 1,
'need': 1, 'neighbor': 1, 'network': 1, 'never': 1, 'news': 1, 'next':
4, 'niece': 2, 'no': 7, 'nod': 1, 'not': 10, 'nothing': 5, 'noticed':
1, 'now': 5, 'o': 2, 'odd': 1, 'of': 34, 'oldemort': 23, 'on': 7, 'onc
e': 3, 'one': 8, 'only': 2, 'open': 1, 'opened': 1, 'opportunity': 1,
'or': 4, 'order': 3, 'ormtail': 1, 'other': 1, 'others': 1, 'ou': 1,
'our': 7, 'ours': 1, 'out': 1, 'outpouring': 1, 'over': 1, 'overhead':
1, 'own': 3, 'p': 6, 'pain': 1, 'pale': 1, 'parted': 1, 'passed': 1,
'people': 2, 'person': 1, 'pius': 1, 'place': 3, 'placing': 1, 'plan
s': 1, 'planted': 1, 'pleasure': 3, 'portion': 1, 'potter': 12, 'powe
r': 1, 'presence': 1, 'prisoner': 1, 'professed': 1, 'protection': 1,
'proud': 1, 'provide': 1, 'pupils': 1, 'put': 1, 'quickly': 1, 'quie
t': 2, 'quite': 1, 'ransport': 1, 'rate': 1, 'reason': 1, 'receive':
2, 'recently': 1, 'red': 3, 'regular': 1, 'regulated': 1, 'remains':
1, 'removing': 1, 'remus': 1, 'repeated': 1, 'repressed': 1, 'requir
e': 1, 'resentfully': 1, 'response': 1, 'rest': 3, 'return': 1, 'revol
ving': 2, 'right': 1, 'rigid': 1, 'rise': 1, 'robes': 1, 'room': 1, 'r
ose': 1, 'rowling': 6, 's': 10, 'said': 11, 'sat': 2, 'saturday': 3,
'saying': 1, 'score': 1, 'scrambled': 1, 'scrimgeour': 2, 'scurried':
1, 'seat': 1, 'second': 1, 'see': 2, 'seem': 1, 'seemed': 5, 'seemingl
y': 2, 'set': 2, 'several': 1, 'shadowed': 1, 'shall': 4, 'she': 8, 's
hock': 1, 'shook': 1, 'shoulders': 2, 'shudder': 1, 'side': 1, 'sidewa
ys': 1, 'silver': 1, 'since': 1, 'sister': 2, 'sitting': 2, 'skin': 1,
'sliding': 1, 'slim': 1, 'slits': 1, 'slowly': 2, 'small': 1, 'snake':
3, 'snape': 3, 'sniggered': 1, 'so': 7, 'soft': 1, 'some': 3, 'somethi
ng': 1, 'son': 1, 'sound': 1, 'sounded': 1, 'source': 1, 'speak': 1,
'speaking': 1, 'spoke': 1, 'spoken': 1, 'squared': 1, 'stared': 1, 'st
aring': 1, 'start': 1, 'startled': 1, 'stif': 1, 'still': 2, 'stoppe
d': 1, 'straight': 1, 'strangely': 1, 'stroked': 1, 'subjugate': 1, 's
ubsided': 1, 'suceeded': 1, 'such': 2, 'sudden': 1, 'sunken': 1, 'surr
ounded': 1, 'sweat': 1, 't': 4, 'table': 9, 'take': 1, 'taken': 1, 'ta
king': 1, 'talking': 1, 'tears': 1, 'tense': 1, 'terrible': 1, 'terrif
ied': 1, 'than': 3, 'that': 17, 'the': 81, 'their': 5, 'them': 4, 'the
n': 2, 'there': 6, 'these': 1, 'they': 6, 'thickness': 1, 'thickness
e': 4, 'thigh': 1, 'thin': 1, 'thing': 1, 'things': 1, 'think': 2, 'th
is': 1, 'those': 4, 'thoughtful': 1, 'throng': 1, 'through': 1, 'thump
ed': 1, 'thwarted': 1, 'tilted': 1, 'to': 37, 'together': 2, 'tone':
1, 'too': 1, 'touch': 2, 'toward': 1, 'transport': 1, 'travels': 1, 't
riumphs': 1, 'true': 1, 'truth': 1, 'turned': 3, 'twisted': 1, 'two':
1, 'ugly': 1, 'unblinking': 1, 'unconscious': 1, 'under': 1, 'understa
nd': 3, 'undo': 1, 'unhappy': 1, 'unless': 1, 'unlike': 1, 'unlikely':
1, 'unoccupied': 1, 'up': 6, 'upon': 1, 'upper': 1, 'us': 1, 'uses':
1, 'v': 23, 'vertical': 1, 'very': 1, 'voice': 3, 'volunteers': 1,
'w': 7, 'wail': 1, 'wand': 9, 'wanted': 1, 'was': 7, 'watched': 1, 'wa
tching': 1, 'waxy': 1, 'way': 1, 'we': 4, 'week': 1, 'welled': 1, 'wen
t': 1, 'were': 2, 'werewolf': 1, 'what': 3, 'when': 1, 'where': 3, 'wh
ich': 1, 'while': 1, 'whispered': 1, 'white': 1, 'who': 4, 'whose': 1,
'why': 1, 'wide': 1, 'widened': 1, 'wife': 2, 'will': 5, 'wiped': 1,
'with': 16, 'withdrew': 1, 'within': 1, 'without': 1, 'wizards': 1, 'w
oman': 1, 'words': 2, 'work': 1, 'wreckers': 1, 'wrist': 1, 'wwwztcpre
pcom': 10, 'y': 9, 'years': 1, 'yellowish': 1, 'you': 12, 'your': 6,
'yours': 1}

In [19]:
```
pip install pyspellchecker
```

```
Requirement already satisfied: pyspellchecker in /opt/anaconda3/lib/py
thon3.11/site-packages (0.8.1)
Note: you may need to restart the kernel to use updated packages.
```

# Question-2 Count how many times non-English words (names, places, spells etc.) were used.

In [32]:
```
import re
from collections import defaultdict
from spellchecker import SpellChecker

# Initialize the spell checker
spell = SpellChecker()

# Preprocessing function: convert text to lowercase, remove punctuation
def preprocess(text):
    text = text.lower()
    words = text.split()
    cleaned_words = [re.sub(r'^[^a-zA-Z\']+', '', re.sub(r'[^a-zA-Z\']+
    return ' '.join(cleaned_words)

# Mapper function
def mapper(text, contractions):
    words = text.split()
    output = defaultdict(int)
    for word in words:
        if word not in contractions and word and spell.unknown([word]):
            output[word] += 1
    return output

# Reducer function
def reducer(map_data):
    output = defaultdict(int)
    for word, count in map_data.items():
        output[word] += count
    return output

# Read the file containing text to be analyzed
with open('file2.txt', 'r') as file:
    text_file = file.read()

# Preprocess the text to clean it
processed_text_file = preprocess(text_file)

# Running MapReduce with contractions passed to mapper
map_data = mapper(processed_text_file, contractions)
reduce_data = reducer(map_data)

# Output in alphabetical order
sorted_dict = dict(sorted(reduce_data.items(), key=lambda x: x[0]))
print(sorted_dict)
```

```
{'accio': 1, 'allen': 1, 'arrior': 1, 'arthur': 1, 'back-to': 1, 'big-
bellied': 1, 'comin': 2, 'confringo': 1, 'didn': 1, 'dragon-fire': 3,
'd'you': 1, 'eal': 1, 'easley': 1, 'ed': 6, 'emer': 1, 'expelliarmus':
1, 'e're': 1, 'ged': 1, 'ground-trembling': 1, 'gut-wrenching': 1, 'ha
grid': 24, 'half-closed': 1, 'hedwig': 1, 'ith': 1, 'i'm': 5, 'j.k':
7, 'lamplit': 1, 'meself': 1, 'muggle': 1, 'oldemort': 9, 'onks': 3,
'rowling': 7, 'selwyn': 1, 'shouldn': 1, 'shunpike': 1, 'spread-eagle
d': 2, 'stan': 2, 'stanley': 1, 'ter': 1, 'thestral': 1, 'vada': 1, 'w
and-free': 1, 'where've': 1, 'white-blue': 1, 'www.ztcprep.com': 10,
'yeh've': 1}
```