DESCRIPTIONS OF EVERY FUNCTION (ALSO IN MAIN PYTHON FILE)

```
from typing import *
from dataclasses import dataclass
import unittest


# put all of your functions and test cases in this file, yay!
#Data Definition: hashtable is a class with a size (int) object, table (list
object), and count (int) object
#Purpose statement: In this class I am creating a HashTable
#Header: this class represents a hashtable, which can create a hash table,
count elements in a hash table, tell whether hash table is full and print the
hashtable
@dataclass
class HashTable:
   size: int
   table: list
   count: int

   # Data Definition: This method just takes in the hashtable and returns
another hashtable
   # Purpose statement: In this method, I am creating a new empty hash table
   # Header: This method creates an empty hash table the size that the user
inputs and returns it
   def create_hash(self):


   # Data Definition: This method just takes in the hashtable and returns an
int counts
   # Purpose statement: In this method, I am counting the amount of elements in
the hash table
   # Header: This method returns the amount of elements in the hash table
   def counts(self):


   # Data Definition: This method just takes in the hashtable and returns None
   # Purpose statement: In this method, I am printing the hash table
   # Header: This method prints the hash table and does not return anything
   def show_hash(self):

   # Data Definition: This method just takes in the hashtable and returns True
or False
   # Purpose statement: In this method, I am returning true if the table is
full and false if it is not
   # Header: This method returns a bool stating whether the table is full or
not
   def full(self):
```

```
# Data Definition: This function just takes in the size of the hash table (int)
and returns another hashtable
# Purpose statement: In this function, I am creating a new hash table
# Header: This function uses the create hash method to create a new empty hash
table the size that the user inputs
def make_hash(size: int) -> HashTable:

# Data Definition: This function just takes in a hashtable and returns an int
# Purpose statement: In this function, I am returning the size of the hash
table
# Header: This function returns the size of an inputted hash table
def hash_size(ht: HashTable) -> int:

# Data Definition: This function just takes in a hashtable and returns an int
# Purpose statement: In this function, I am counting the amount of elements in
a hash table and returning it as an int
# Header: This function uses the counts method to count the amount of elements
in a hash table and returns it as an int
def hash_count(ht: HashTable) -> int:

# Data Definition: This function just takes in a word and returns its hash
index value as an integer
# Purpose statement: In this funciton, calculating the hash index of a word
with horners method
# Header: This function returns the hash value of a word
def quadratic_hash(word):

# Data Definition: This function just takes in a hashtable and a word (string)
and returns a bool
# Purpose statement: In this function, I am checking whether the word ("key")
is in the hashtable and returning a bool to tell whether it is or isnt
# Header: This function returns true if the word is in the hashtable and false
if not
def has_key(ht: HashTable, word: str) -> bool:

# Data Definition: This function just takes in a hashtable returns a list of
ints representing page numbers
# Purpose statement: In this method, I am returning the page numbers in a list
of a word that is in the hash table
# Header: This function returns the page numbers of a looked up word ("key")
def hash_keys(ht: HashTable) -> List[str]:

# Data Definition: This function just takes in a file and returns a list
# Purpose statement: This function reads all of the stop_words from the
stopwords file and returns a list of them
# Header: This function returns all of the stop words from a file into a list
for the concordance funciton
def read_stop_words(file_path):
```

```
# Data Definition: This function just takes in a file and returns a list
# Purpose statement: This function reads all of the lines from the text file
and returns a string of all of the text
# Header: This function concatonates a list of words in the file into a string
to be cleaned
def read_test(file_path):


# Data Definition: This function just takes in a string and returns a list
# Purpose statement: This function takes the apostrophes out of string and
lowercases/takes out any punctuation
# Header: This function cleans the string and turns it into a list if alpha is
true or if a newline
def clean_text(text: str):

# Data Definition: This function just takes in two lists (stop words and text)
and returns a hashtable
# Purpose statement: This function makes a hash table for both the stop_words
and the text words
#it then checks the text hash against the stop_words hash to not include the
stop words into the hash
# and then it adds a line number for every newline break character
#it then creates a new hash table with the values
# Header: This function essentially makes a hashtable from the words in the
text file and excludes words from the stop_words file
def make_concordance(stop_words: List[str], text: List[str]) -> HashTable:

# Data Definition: This function just takes in a hash table and a string of the
output file and returns None
# Purpose statement: This function writes the hash table values to the
concordance_output file
# Header: This function writes the hash table values to the concordance_output
file
def generate_concordance_file(ht: HashTable, output_file_path: str) -> None:

# Data Definition: This function just takes in a hash table and returns a hash
table
# Purpose statement: This function doubles the size of the hash table when it
is full
# Header: This function doubles the size of the hash table when it is full
def grow_hash(ht: HashTable) -> HashTable:

# Data Definition: This function just takes in two hash tables hash table and
returns a hash table
# Purpose statement: This function inserts words into the hashtable in the
correct place
# Header: This function inserts words into the hashtable in the correct place
def rehash(ht: HashTable, temp_ht: HashTable) -> HashTable:
```

```
# Data Definition: This function just takes in a hash table, a word, and a list
of line numbers and returns None
# Purpose statement: This function adds the word and line numbers to the
hashtable
# Header: This function adds the word and line numbers to the hashtable
#take in hash table word and line
#if hash is full, grow and rehash
#else
#calculate index
#quadratically probe and add when ht[index] is None
def add(ht: HashTable, word: str, line: int) -> None:

# Data Definition: This function just takes in a hash table and a word and
returns a list of ints
# Purpose statement: This function uses the key to return the page numbers
associated with that key
# Header: This function uses the key to return the page numbers associated with
that key
def lookup(ht: HashTable, word: str) -> List[int]:

#this code allows the user to input the name of the input text file which will
then output to concordance_output.txt
#INSTRUCTIONS ON HOW TO CALL PROGRAM
#1. input name of input file when prompted in terminal (without adding '.txt'
(the main input file in this folder is 'text.txt')
#2. concordance will be outputted to concordance_output.txt
print("INPUT TEXTFILE HERE (DON'T INCLUDE '.txt'): ")
userinput = input() + ".txt"  #input text file
stringoftext = read_test(userinput) #processing text file
cleaned_text = clean_text(stringoftext) #cleaning text file
stop_words = read_stop_words('stop_words.txt') #get stop words
concordance = make_concordance(stop_words, cleaned_text) #input text and stop
words into concordance file
generate_concordance_file(concordance, 'concordance_output.txt') #output
concordance to concordance_output.txt
```

INSTRUCTIONS ON HOW TO CALL PROGRAM

1. input name of input file when prompted in terminal (without adding '.txt' (the main input file in this folder is 'text.txt')
2. concordance will be outputted to concordance_output.txt

DESCRIPTION OF RUNNING FILE ON OVER ONE MEGABYTE

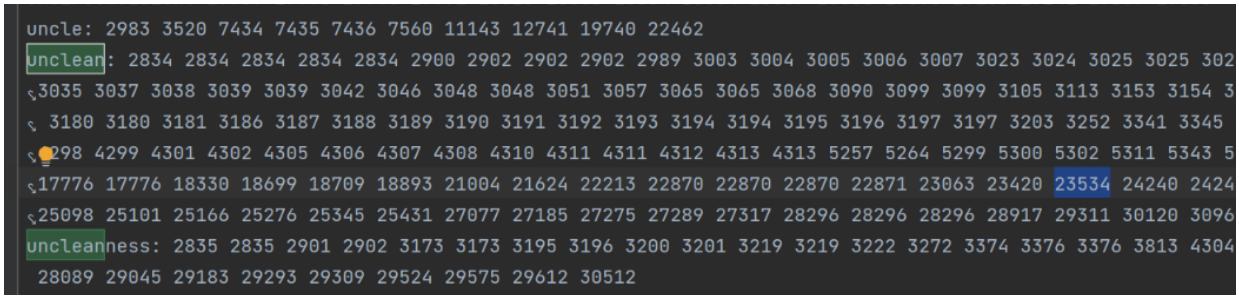
I ran a bible (4.44 MB) https://www.o-bible.com/download/kjv.txt (took 9.45 seconds to run)
total length of output: 12125 lines

RANDOM 5 LINES I CHOSE: (Lines 23530 - 23534)
Mat12:39 But he answered and said unto them, An evil and adulterous generation seeketh after a sign; and there shall no sign be given to it, but the sign of the prophet Jonas:
Mat12:40 For as Jonas was three days and three nights in the whale's belly; so shall the Son of man be three days and three nights in the heart of the earth.
Mat12:41 The men of Nineveh shall rise in judgment with this generation, and shall condemn it: because they repented at the preaching of Jonas; and, behold, a greater than Jonas is here.
Mat12:42 The queen of the south shall rise up in the judgment with this generation, and shall condemn it: for she came from the uttermost parts of the earth to hear the wisdom of Solomon; and, behold, a greater than Solomon is here.
Mat12:43 When the unclean spirit is gone out of a man, he walketh through dry places, seeking rest, and findeth none.

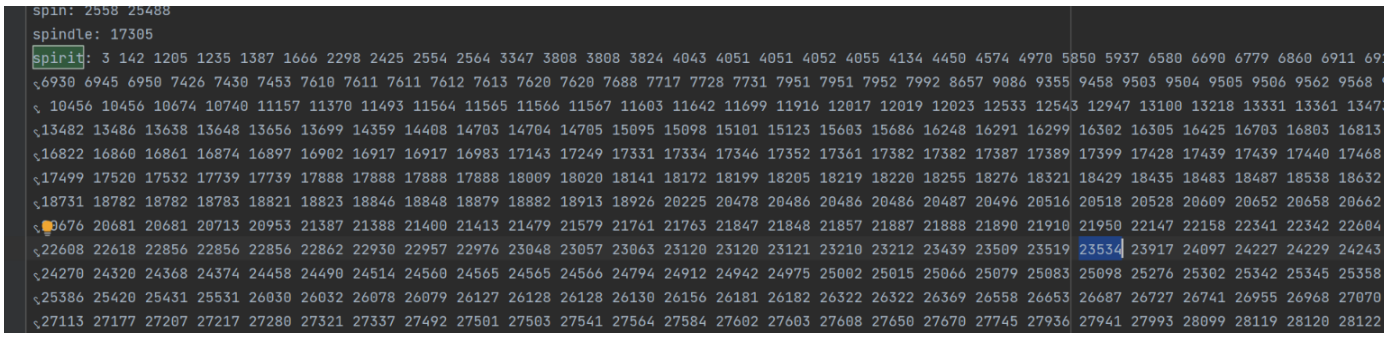
ONE LINE: (Line 23534)
Mat12:43 When the unclean spirit is gone out of a man, he walketh through dry places, seeking rest, and findeth none.

When

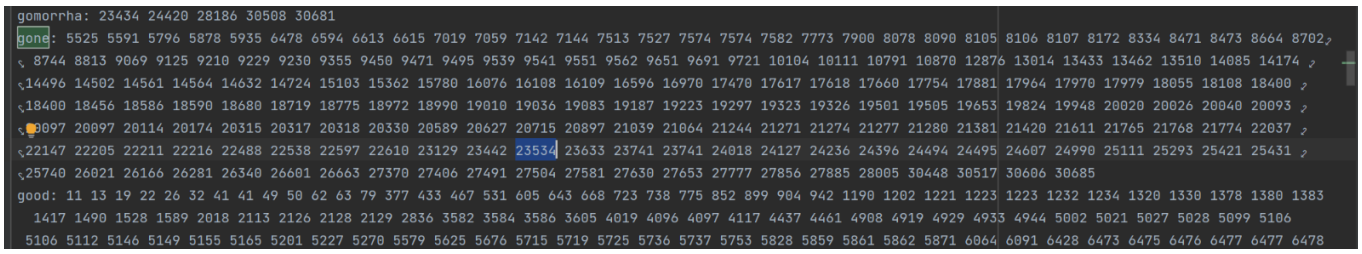
Unclean



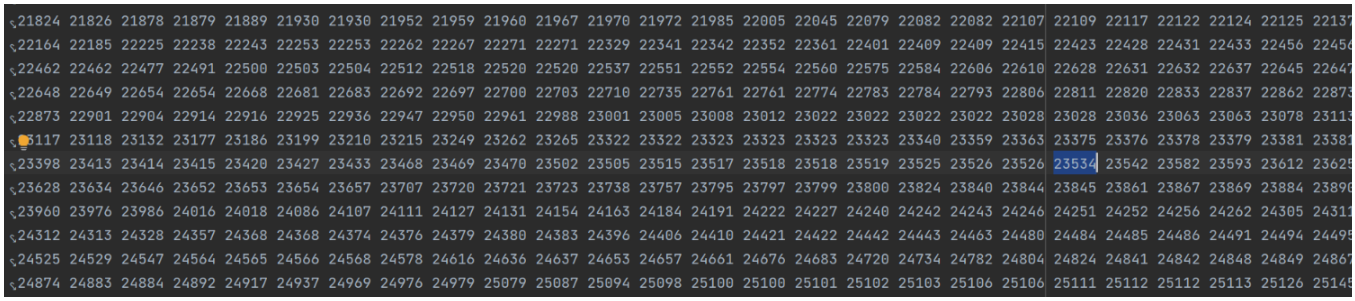
Spirit



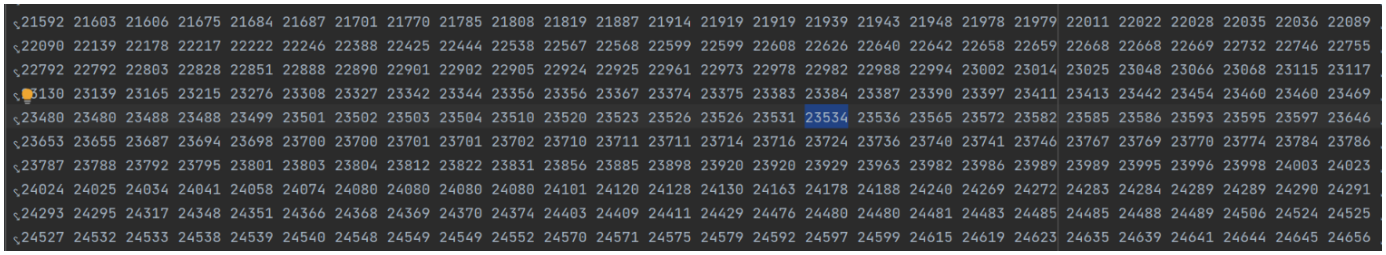
Gone



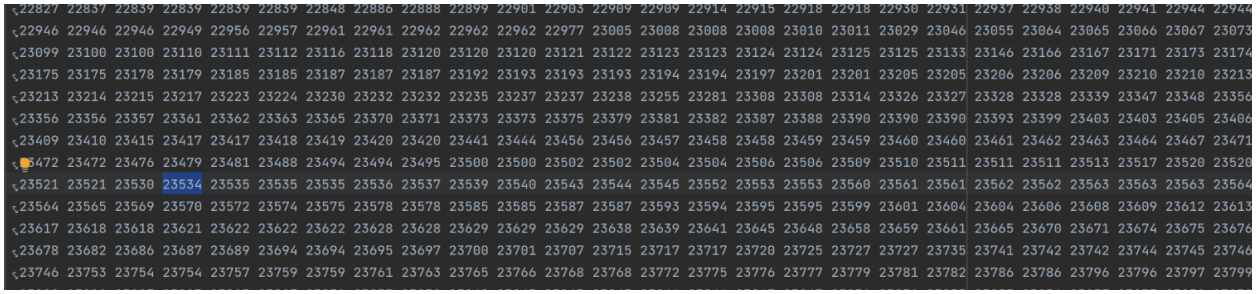
Out



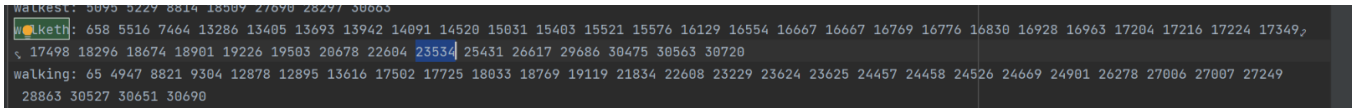
Man



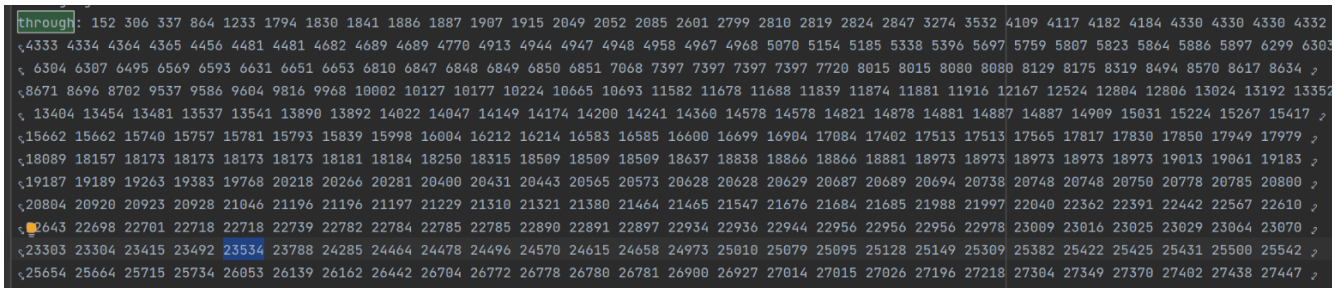
He



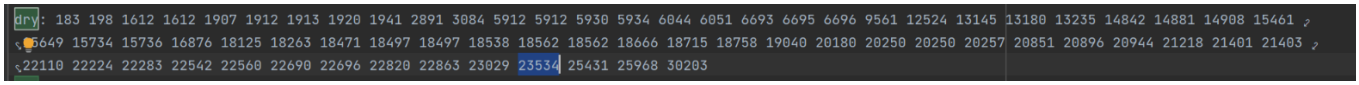
Walketh



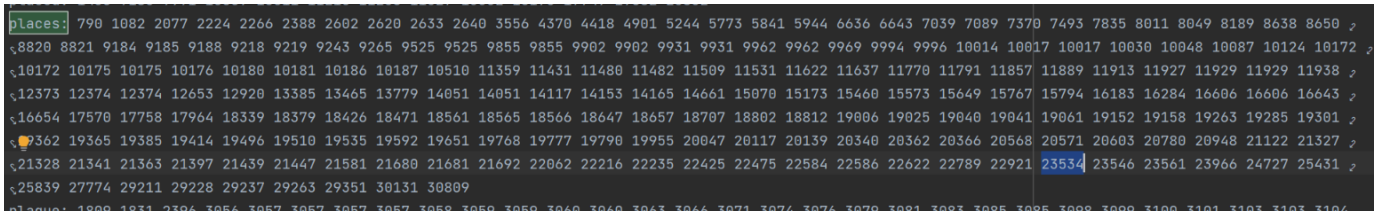
through



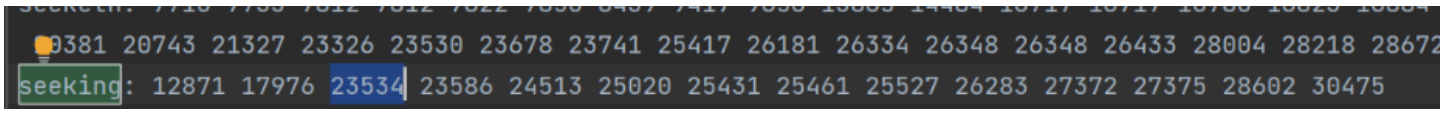
Dry



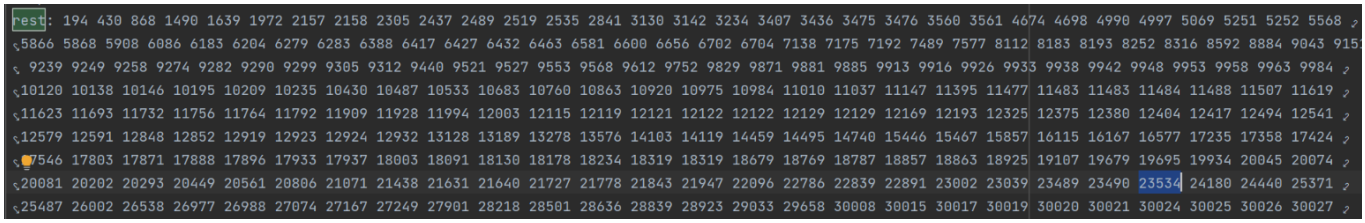
Places



Seeking



rest



Findeth

```
 26825 26831 26833 26906 27164 27552 27745 28111 28114 28176 28962 29044 29829 30032 30848 31009
findest: 907
findeth: 95 13662 16062 16470 16639 16639 16780 16895 16925 16925 16996 17007 17487 20315 22287 23458 23534 23535 24096 24793 25432 26087 26089 26091 26226
finding: 96 13063 18801 25431 27045 27588 27668 27670 28244 30102
```

None

```
non: 10564
none: 579 792 1160 1162 1205 1212 1221 1236 1722 1758 1768 1814 1840 1948 1975 1976 2161 2518 3259 3348 3401 3497 3532 3543 3562 3563 3601 3979 4377 4658 4731
4974 4980 5041 5045 5062 5128 5499 5644 5679 5796 5838 5952 5952 6026 6062 6087 6094 6096 6099 6103 6105 6106 6117 6122 6131 6170 6192 7054 7112 7113 7196 7244
7244 7297 7444 7534 7783 7797 7797 8204 8364 8377 8383 8492 8646 8830 9047 9102 9173 9273 9478 9665 9688 9768 9773 9806 9814 9818 9820 10003 10031 10218 10795
10885 11002 11181 11208 11377 11386 11413 11512 11595 11613 11664 11677 12218 12384 12712 12766 12879 12896 12906 13095 13129 13293 13349 13546 13642 13732
13734 13900 13999 14058 14083 14085 14161 14217 14235 14256 14378 14412 14483 14657 14692 14722 14724 14957 14962 14989 15047 15088 15190 15230 15294 15713
15769 16257 16427 16432 16454 16488 17586 17687 17768 17768 17770 17866 17936 17961 17987 18076 18076 18315 18317 18321 18470 18479 18479 18479 18504 18504
18520 18554 18568 18569 18569 18577 18581 18584 18585 18597 18597 18609 18611 18611 18616 18666 18693 18768 18806 18871 18873 18873 18894 18928 19033 19051
19154 19187 19189 19199 19209 19210 19223 19287 19311 19454 19500 19676 19679 19682 19812 19813 19839 19874 19994 20019 20026 20026 20074 20115 20134 20171
20177 20197 20200 20276 20314 20316 20319 20329 20333 20356 20452 20590 20593 20604 20710 20769 20798 20858 21008 21246 21298 21310 21321 21343 21476 21478
1758 21771 21874 21911 21970 21990 22038 22054 22083 22093 22117 22168 22187 22249 22262 22340 22427 22431 22519 22602 22621 22626 22643 22668 22709 22710
22712 22717 22822 22828 22828 22835 22848 22974 22995 23120 23534 23641 23781 24116 24478 24608 24706 24707 24811 24956 25091 25092 25431 25526 25579 25709
25724 26281 26349 26393 26725 26733 26773 26796 26912 27036 27123 27194 27202 27328 27576 27652 27794 27809 27816 27847 27851 28003 28004 28004 28005 28038
```

ANIMAL:

panda