

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 concatenates a list of words in the file into a string
to be cleaned
def read_text(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_text(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

```
uncle: 2983 3520 7434 7435 7436 7560 11143 12741 19740 22462  
Unclear: 2834 2834 2834 2834 2834 2900 2902 2902 2902 2989 3003 3004 3005 3006 3007 3023 3024 3025 3025 302  
`3035 3037 3038 3039 3039 3042 3046 3048 3048 3051 3057 3065 3065 3068 3090 3099 3099 3105 3113 3113 3153 3154 3  
` 3180 3180 3181 3186 3187 3188 3189 3190 3191 3192 3193 3194 3194 3195 3195 3196 3197 3197 3203 3252 3341 3345  
` 298 4299 4301 4302 4305 4306 4307 4308 4310 4311 4311 4312 4313 4313 5257 5264 5299 5300 5302 5311 5343 5  
` 17776 17776 18330 18699 18709 18893 21004 21624 22213 22870 22870 22870 22871 23063 23420 23534 24240 2424  
` 25098 25101 25166 25276 25345 25431 27077 27185 27275 27289 27317 28296 28296 28296 28917 29311 30120 3096  
unclearness: 2835 2835 2901 2902 3173 3173 3195 3196 3200 3201 3219 3219 3222 3272 3374 3376 3376 3813 4304  
28089 29045 29183 29293 29309 29524 29575 29612 30512
```

Spirit

```
spin: 2558 25488
spindle: 17305
spirit: 3 142 1205 1235 1387 1666 2298 2425 2554 2564 3347 3808 3808 3824 4043 4051 4051 4052 4055 4134 4450 4574 4970 5850 5937 6580 6690 6779 6860 6911 69
< 6938 6945 6950 7426 7430 7453 7610 7611 7612 7613 7620 7620 7688 7717 7728 7731 7951 7951 7952 7992 8857 9086 9355 9458 9503 9504 9505 9506 9562 9568
< 10456 10456 10674 10740 11157 11370 11493 11564 11565 11566 11567 11603 11642 11699 11916 12017 12019 12023 12533 12543 12947 13100 13218 13331 13361 1347
< 13482 13486 13638 13648 13656 13699 14359 14408 14703 14704 14705 15095 15098 15101 15123 15603 15686 16248 16291 16299 16302 16305 16425 16703 16803 16813
< 16822 16860 16861 16874 16897 16902 16917 16917 16983 17143 17249 17331 17334 17346 17352 17361 17382 17387 17389 17399 17428 17439 17439 17440 17468
< 17499 17520 17532 17739 17739 17888 17888 17888 18009 18020 18141 18172 18199 18205 18219 18220 18255 18276 18321 18429 18435 18483 18487 18538 18632
< 18731 18782 18782 18821 18823 18846 18848 18879 18882 18913 18926 20225 20478 20486 20486 20486 20486 20487 20496 20516 20518 20528 20609 20652 20658 20662
< 20676 20681 20681 20713 20953 21387 21388 21400 21413 21479 21579 21761 21763 21847 21848 21857 21887 21888 21890 21910 21950 22147 22158 22341 22342 22684
< 22608 22618 22856 22856 22862 22930 22957 22976 23048 23057 23063 23120 23120 23210 23212 23439 23509 23519 23534 23917 24097 24227 24229 24243
< 24270 24320 24368 24374 24458 24490 24514 24560 24565 24566 24566 24794 24912 24942 24975 25002 25015 25066 25079 25083 25098 25276 25302 25342 25345 25358
< 25386 25420 25431 25531 26030 26032 26078 26079 26127 26128 26128 26130 26156 26181 26182 26322 26322 26369 26558 26653 26687 26727 26741 26955 26968 27070
< 27113 27177 27207 27217 27280 27321 27337 27492 27501 27503 27541 27564 27584 27602 27603 27608 27650 27670 27745 27936 27941 27993 28099 28119 28120 28122
```

Gone

```
gomrrha: 23434 24420 28186 30508 30681
gone: 5525 5591 5796 5878 5935 6478 6594 6613 6615 7019 7059 7142 7144 7513 7527 7574 7574 7582 7773 7900 8078 8090 8105 8106 8107 8172 8334 8471 8473 8664 8702,
< 8744 8813 9069 9125 9210 9229 9230 9355 9450 9471 9495 9539 9541 9551 9562 9651 9691 9721 10104 10111 10791 10870 12876 13014 13433 13462 13510 14085 14174 ,
< 14496 14502 14561 14564 14632 14724 15103 15362 15780 16076 16108 16109 16596 16970 17478 17617 17618 17660 17754 17881 17964 17970 17979 18055 18108 18408 ,
< 18400 18456 18586 18590 18680 18719 18775 18972 18990 19010 19036 19083 19187 19223 19297 19323 19326 19501 19505 19653 19824 19948 20020 20026 20040 20093 ,
< 20897 20897 20814 20174 20315 20317 20318 20330 20589 20627 20715 20897 21039 21064 21244 21271 21274 21277 21280 21381 21420 21611 21765 21768 21774 22037 ,
< 22147 22205 22211 22216 22488 22538 22597 22610 23129 23442 23534 23633 23741 23741 24018 24127 24256 24396 24494 24495 24607 24990 25111 25293 25421 25431 ,
< 25740 26021 26166 26281 26340 26601 26663 27370 27406 27491 27504 27581 27630 27653 27777 27856 27885 28005 30488 30517 30606 30685
good: 11 13 19 22 26 32 41 41 49 50 62 63 79 377 433 467 531 605 643 668 723 738 775 852 899 984 942 1198 1202 1221 1223 1232 1234 1320 1330 1378 1380 1383
1417 1490 1528 1589 2018 2113 2126 2128 2129 2836 3582 3584 3586 3605 4019 4096 4097 4117 4437 4461 4908 4919 4929 4933 4944 5002 5021 5027 5028 5099 5106
5106 5112 5146 5149 5155 5165 5201 5227 5270 5579 5625 5676 5715 5719 5725 5736 5737 5753 5828 5859 5861 5862 5871 6064 6091 6428 6473 6475 6477 6477 6478
```

Out

```
< 21824 21826 21878 21879 21889 21930 21930 21952 21959 21968 21968 21967 21970 21972 21985 22005 22005 22045 22079 22082 22082 22107 22109 22117 22122 22124 22125 22137
< 22164 22185 22225 22238 22243 22253 22253 22262 22267 22271 22329 22341 22342 22352 22361 22401 22409 22409 22415 22423 22428 22431 22433 22456 22456
< 22462 22462 22477 22491 22500 22504 22512 22518 22520 22520 22537 22551 22552 22554 22560 22575 22584 22606 22610 22628 22631 22632 22637 22645 22647
< 22648 22649 22654 22654 22668 22681 22683 22692 22697 22708 22708 22763 22710 22735 22761 22761 22774 22784 22793 22806 22811 22820 22833 22837 22862 22873
< 22873 22901 22904 22914 22916 22925 22936 22947 22950 22961 22988 23001 23005 23008 23012 23022 23022 23022 23022 23028 23028 23036 23063 23078 23113
< 23117 23118 23132 23177 23186 23199 23210 23215 23249 23262 23265 23322 23322 23323 23323 23323 23340 23359 23363 23375 23376 23378 23379 23381 23381
< 23398 23413 23414 23415 23420 23427 23433 23468 23469 23470 23502 23505 23515 23517 23518 23518 23519 23525 23526 23526 23534 23542 23582 23593 23612 23625
< 23628 23634 23646 23652 23653 23654 23657 23707 23720 23721 23723 23738 23757 23795 23797 23799 23800 23824 23840 23844 23845 23861 23867 23869 23884 23896
< 23960 23976 23986 24016 24086 24107 24111 24127 24131 24154 24163 24184 24191 24222 24227 24240 24242 24243 24246 24251 24252 24256 24262 24305 24312
< 24312 24313 24328 24357 24368 24368 24374 24376 24379 24388 24388 24396 24406 24406 24421 24422 24422 24444 24463 24480 24484 24485 24486 24491 24494 24495
< 24525 24529 24547 24564 24565 24566 24568 24578 24616 24636 24637 24653 24657 24661 24676 24683 24720 24734 24782 24804 24824 24841 24842 24848 24849 24867
< 24874 24883 24884 24892 24917 24937 24969 24976 24979 25079 25087 25094 25098 25100 25100 25101 25102 25103 25106 25106 25111 25112 25112 25113 25126 25145
```

Man

```
< 21592 21603 21606 21675 21684 21687 21701 21770 21785 21808 21819 21887 21914 21919 21919 21939 21943 21948 21978 21979 22011 22022 22028 22035 22036 22089
< 22090 22139 22178 22217 22222 22246 22388 22425 22444 22538 22567 22568 22599 22608 22626 22640 22642 22658 22659 22668 22668 22669 22732 22746 22755
< 22792 22792 22803 22828 22851 22888 22890 22901 22902 22905 22924 22925 22961 22973 22978 22982 22984 23002 23014 23025 23048 23066 23068 23115 23117
< 23130 23139 23165 23215 23276 23308 23327 23342 23344 23356 23356 23367 23374 23375 23383 23384 23387 23390 23397 23411 23413 23442 23454 23460 23460 23469
< 23480 23480 23488 23488 23499 23501 23502 23503 23504 23510 23520 23523 23526 23526 23531 23534 23536 23565 23572 23582 23585 23586 23593 23595 23597 23646
< 23653 23655 23687 23694 23698 23700 23701 23701 23702 23710 23711 23711 23714 23714 23724 23736 23740 23741 23746 23767 23770 23774 23784 23786
< 23787 23788 23792 23795 23801 23803 23804 23812 23822 23831 23856 23885 23898 23920 23929 23963 23982 23986 23989 23995 23996 23998 24003 24023
< 24024 24025 24034 24041 24058 24074 24080 24080 24080 24101 24120 24128 24130 24163 24178 24188 24240 24269 24272 24283 24284 24289 24289 24290 24291
< 24293 24295 24317 24348 24351 24366 24368 24369 24370 24374 24403 24409 24411 24429 24476 24480 24480 24481 24483 24485 24485 24488 24489 24506 24524 24525
< 24527 24532 24533 24538 24539 24540 24548 24549 24549 24552 24570 24571 24575 24579 24592 24597 24599 24615 24619 24623 24635 24639 24641 24644 24645 24656
```

He

z_22827	22837	22839	22839	22839	22848	22886	22888	22899	22901	22903	22909	22914	22915	22918	22918	22930	22931	22937	22938	22940	22941	22944	22946				
z_22946	22946	22946	22949	22956	22957	22961	22961	22962	22962	22962	22977	23005	23008	23008	23008	23010	23011	23029	23046	23055	23064	23065	23066	23067	23073		
z_23099	23100	23100	23110	23111	23112	23116	23118	23120	23120	23120	23121	23122	23123	23123	23124	23125	23125	23133	23146	23166	23167	23171	23175	23174			
z_23175	23175	23178	23179	23185	23185	23187	23187	23187	23192	23193	23193	23194	23194	23194	23197	23201	23201	23205	23205	23206	23206	23209	23210	23210	23213		
z_23213	23214	23215	23217	23223	23224	23230	23232	23232	23235	23237	23237	23238	23238	23308	23314	23326	23327	23328	23328	23339	23347	23348	23348	23356			
z_23356	23356	23357	23361	23362	23363	23365	23370	23371	23373	23373	23375	23379	23381	23382	23387	23388	23390	23390	23390	23393	23399	23403	23405	23406			
z_23409	23410	23415	23417	23417	23418	23419	23420	23420	23441	23444	23456	23456	23457	23457	23458	23458	23459	23459	23460	23460	23461	23462	23463	23464	23467	23471	
z_23472	23472	23476	23479	23481	23488	23494	23494	23495	23500	23500	23502	23502	23504	23504	23566	23506	23509	23510	23511	23511	23511	23511	23511	23511	23517	23520	23528
z_23521	23521	23530	23534	23535	23535	23535	23536	23537	23539	23540	23543	23544	23545	23552	23553	23553	23560	23561	23561	23562	23562	23563	23563	23563	23563		
z_23564	23565	23569	23570	23570	23572	23574	23575	23578	23588	23585	23587	23587	23593	23594	23595	23595	23598	23599	23601	23604	23604	23606	23608	23612	23613		
z_23617	23618	23621	23621	23622	23622	23628	23628	23629	23629	23629	23638	23639	23641	23645	23648	23658	23659	23661	23665	23670	23671	23674	23675	23676			
z_23678	23682	23686	23687	23689	23694	23694	23695	23697	23700	23701	23707	23715	23717	23717	23720	23725	23727	23727	23735	23741	23742	23742	23744	23745	23746		
z_23746	23753	23754	23754	23757	23759	23759	23761	23763	23765	23766	23768	23768	23772	23775	23776	23777	23779	23779	23781	23782	23786	23786	23796	23796	23797	23799	

Walketh

```
walktest: 5095 5529 8814 18509 27698 28297 30663  
walketh: 658 5516 7464 13286 13405 13693 13942 14091 14520 15031 15403 15521 15576 16129 16554 16667 16667 16769 16776 16830 16928 16963 17204 17216 17224 17349,  
17498 18296 18674 18901 19226 19503 20678 22604 23534 25431 26617 29686 30475 30563 30720  
walking: 65 4947 8821 9304 12878 12895 13616 17502 17725 18033 18769 19119 21834 22608 23229 23624 23625 24457 24458 24526 24669 24901 26278 27006 27007 27249  
28863 30527 30651 30690
```

through

through	152	306	337	864	1233	1794	1830	1841	1886	1887	1907	1915	2049	2052	2085	2601	2799	2810	2819	2824	2847	3274	3532	4109	4117	4182	4184	4330	4330	4332			
↳	4333	4334	4364	4365	4456	4481	4481	4682	4689	4689	4770	4913	4944	4947	4948	4958	4967	4968	5070	5154	5185	5338	5396	5697	5759	5807	5823	5864	5886	5897	6299	6303	
↳	6304	6307	6495	6569	6593	6631	6651	6653	6810	6847	6848	6849	6850	6851	7068	7397	7397	7397	7397	7720	8015	8015	8080	8080	8129	8175	8319	8494	8570	8617	8634	;	
↳	8671	8696	8702	9537	9586	9604	9816	9968	10068	10127	10177	10224	10665	10693	11582	11678	11688	11839	11874	11881	11916	12167	12524	12804	12806	13024	13192	13352	;				
↳	13040	13044	13481	13537	13541	13890	13892	14022	14047	14149	14174	14200	14241	14360	14578	14821	14878	14881	14887	14887	14909	15031	15224	15267	15417	,	,	,	,	,			
↳	15662	15662	15740	15757	15781	15793	15989	15998	16004	16212	16214	16583	16585	16600	16699	16904	17084	17402	17513	17513	17565	17817	17830	17850	17949	17979	,	,	,	,	,		
↳	18089	18157	18173	18173	18173	18173	18181	18184	18250	18315	18509	18509	18509	18637	18838	18866	18866	18881	18973	18973	18973	18973	18973	18973	18973	19013	19061	19183	,	,	,	,	,
↳	19187	19189	19263	19383	19768	20218	20266	20281	20400	20431	20443	20565	20573	20628	20628	20629	20687	20689	20694	20738	20748	20748	20750	20778	20785	20800	,	,	,	,	,		
↳	20804	20920	20923	20928	21046	21196	21197	21229	21310	21321	21380	21464	21465	21547	21676	21684	21685	21988	21997	22040	22362	22391	22442	22567	22610	,	,	,	,	,			
↳	26463	22998	22701	22718	22718	22739	22782	22784	22785	22785	22890	22891	22897	22934	22936	22944	22956	22956	22956	22978	23009	23016	23025	23029	23064	23070	,	,	,	,	,		
↳	23303	23304	23415	23492	23534	23788	24285	24464	24478	24496	24570	24615	24658	24733	25010	25079	25095	25128	25149	25309	25382	25422	25425	25431	25500	25542	,	,	,	,	,		
↳	25654	25664	25715	25734	26053	26139	26162	26442	26704	26772	26778	26780	26781	26900	26927	27014	27015	27026	27196	27218	27304	27349	27378	27402	27448	27447	,	,	,	,	,		

Dry

```
dry: 183 198 1612 1612 1997 1912 1913 1926 1941 2891 3084 5912 5912 5930 5934 6044 6051 6693 6695 6696 9561 12524 13145 13180 13235 14842 14881 14908 15461 ,  
§ 649 15734 15736 16876 18125 18263 18471 18497 18497 18538 18562 18562 18666 18715 18758 19040 20180 20250 20250 20257 20851 20896 20944 21218 21401 21403 ,  
§ 22110 22224 22283 22542 22560 22690 22696 22828 22863 23029 23534 25431 25968 30203
```

Places

<code>places:</code>	790	1082	2077	2224	2266	2388	2602	2620	2633	2640	3556	4370	4418	4901	5244	5773	5841	5944	6636	6643	7039	7089	7370	7493	7835	8011	8049	8189	8638	8650	,
<code> \\$8820</code>	8821	9184	9185	9188	9218	9219	9243	9265	9525	9825	9855	9855	9982	9982	9931	9931	9962	9962	9969	9994	9996	10014	10017	10017	10050	10048	10087	10124	10172	,	
<code> \\$10172</code>	10175	10175	10176	10188	10181	10186	10187	10510	11359	11431	11480	11482	11509	11531	11622	11637	11770	11791	11857	11889	11913	11927	11929	11929	11938	,					
<code> \\$12373</code>	12374	12374	12653	12928	13385	13465	13779	14051	14051	14117	14153	14165	14661	15070	15173	15460	15573	15649	15767	15794	16183	16284	16686	16686	16643	,					
<code> \\$16654</code>	17578	17758	17964	18339	18379	18426	18471	18561	18565	18566	18647	18657	18707	18802	18812	19006	19025	19040	19041	19061	19152	19158	19263	19285	19301	,					
<code> \\$19362</code>	19362	19365	19385	19414	19496	19510	19535	19592	19651	19768	19777	19798	19955	20047	20117	20139	20340	20362	20366	20568	20571	20603	20780	20948	21122	21327	,				
<code> \\$21328</code>	21341	21363	21397	21439	21447	21581	21680	21681	21692	22062	22216	22235	22425	22475	22584	22586	22622	22789	22921	23534	23546	23561	23966	24727	25431	,					
<code> \\$25839</code>	27774	29211	29228	29237	29263	29351	30131	30809																						,	
<code>places:</code>	1899	1831	2394	3056	3057	3057	3057	3057	3058	3059	3059	3060	3060	3060	3063	3064	3071	3074	3074	3079	3081	3083	3085	3090	3099	3100	3101	3103	3103	,	

Seeking

```
SECRETEN: 7710 7730 7812 7812 7822 7830 8487 7417 7838 10803 14484 15717 16717 16730 16820 16884  
9381 20743 21327 23326 23530 23678 23741 25417 26181 26334 26348 26348 26433 28004 28218 28672  
seeking: 12871 17976 23534 23586 24513 25020 25431 25461 25527 26283 27372 27375 28602 30475
```

rest

```
rest: 194 430 868 1490 1639 1972 2157 2158 2305 2437 2489 2519 2535 2841 3130 3142 3234 3407 3436 3475 3476 3568 3561 4674 4698 4998 4997 5069 5251 5252 5568 ,  
5866 5868 5908 6086 6183 6204 6279 6283 6388 6417 6427 6432 6463 6581 6600 6656 6702 6704 7138 7175 7192 7489 7577 8112 8183 8193 8252 8316 8592 8884 9043 9151  
9239 9249 9258 9274 9282 9290 9299 9305 9312 9440 9521 9527 9553 9568 9612 9752 9829 9871 9881 9885 9913 9916 9926 9933 9938 9942 9948 9953 9958 9963 9984 ,  
10120 10138 10146 10195 10209 10235 10430 10487 10533 10683 10760 10863 10920 10975 10984 11010 11037 11147 11395 11477 11483 11484 11488 11507 11619 ,  
11623 11693 11732 11756 11764 11792 11909 11928 11994 12003 12115 12119 12121 12122 12129 12129 12169 12193 12325 12375 12380 12404 12417 12494 12541 ,  
12579 12591 12848 12852 12919 12923 12924 12932 13128 13189 13278 13576 14103 14119 14459 14495 14740 15446 15467 15857 16115 16167 16577 17235 17358 17424 ,  
17546 17803 17871 17888 17896 17933 17937 18003 18091 18130 18178 18234 18319 18319 18679 18769 18878 18857 18863 18925 19107 19679 19695 19934 20045 20074 ,  
20081 20202 20293 20449 20561 20886 21071 21438 21631 21648 21727 21778 21843 21947 22096 22786 22839 22891 23082 23639 23489 23496 23534 24180 24440 25371 ,  
25487 26002 26538 26977 26988 27074 27167 27249 27901 28218 28501 28636 28839 28923 29033 29658 30008 30015 30017 30019 30020 30021 30024 30025 30026 30027 ,
```

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

```
nom: 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 5045 5062 5128 5499 5644 5679 5796 5838 5952 5952 6026 6062 6087 6094 6096 6099 6103 6106 6117 6122 6131 6178 6192 7054 7112 7113 7196 7244,
` 7244 7297 7444 7534 7783 7797 7797 8264 8364 8377 8383 8492 8646 8830 9047 9102 9173 9273 9478 9685 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 17770 17866 17936 17961 17987 18076 18076 18315 18317 18321 18470 18479 18479 18479 18504 18504 ,
` 18520 18554 18568 18569 18569 18577 18581 18584 18584 18597 18597 18609 18611 18611 18616 18666 18693 18768 18806 18866 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 20684 20718 20769 20798 20858 21088 21246 21298 21310 21321 21343 21476 21478 ,
` 21758 21771 21874 21911 21970 21990 22838 22854 22883 22893 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 28005 28038 ,
```

ANIMAL:

panda