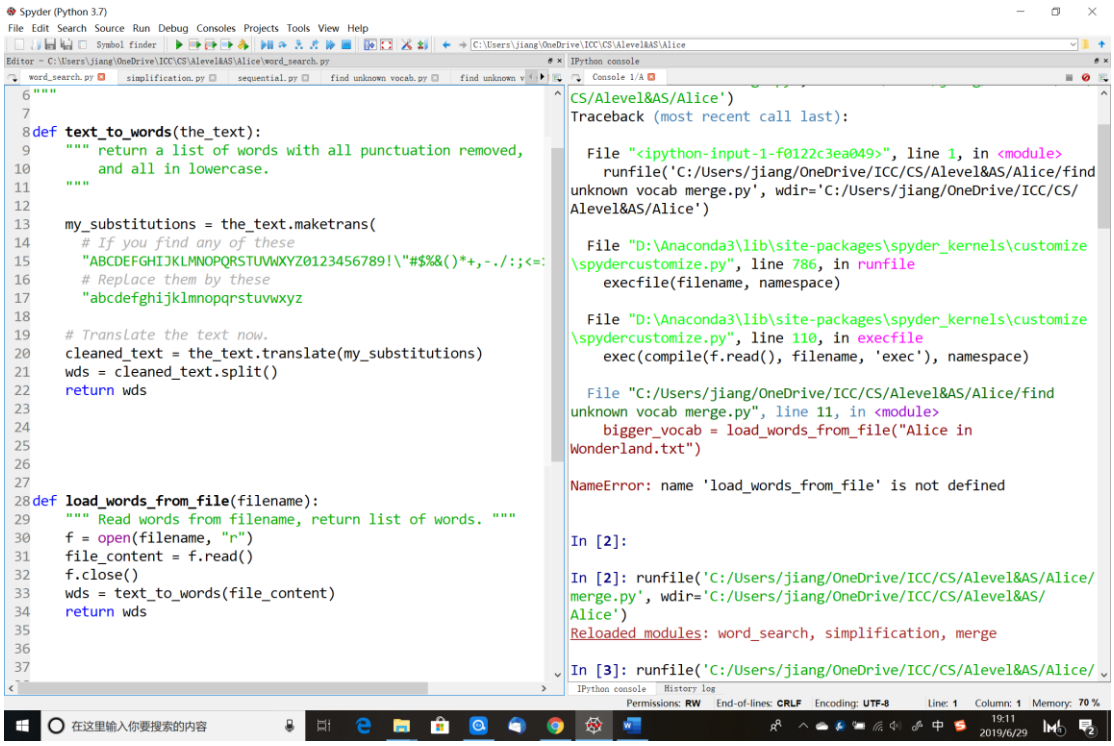
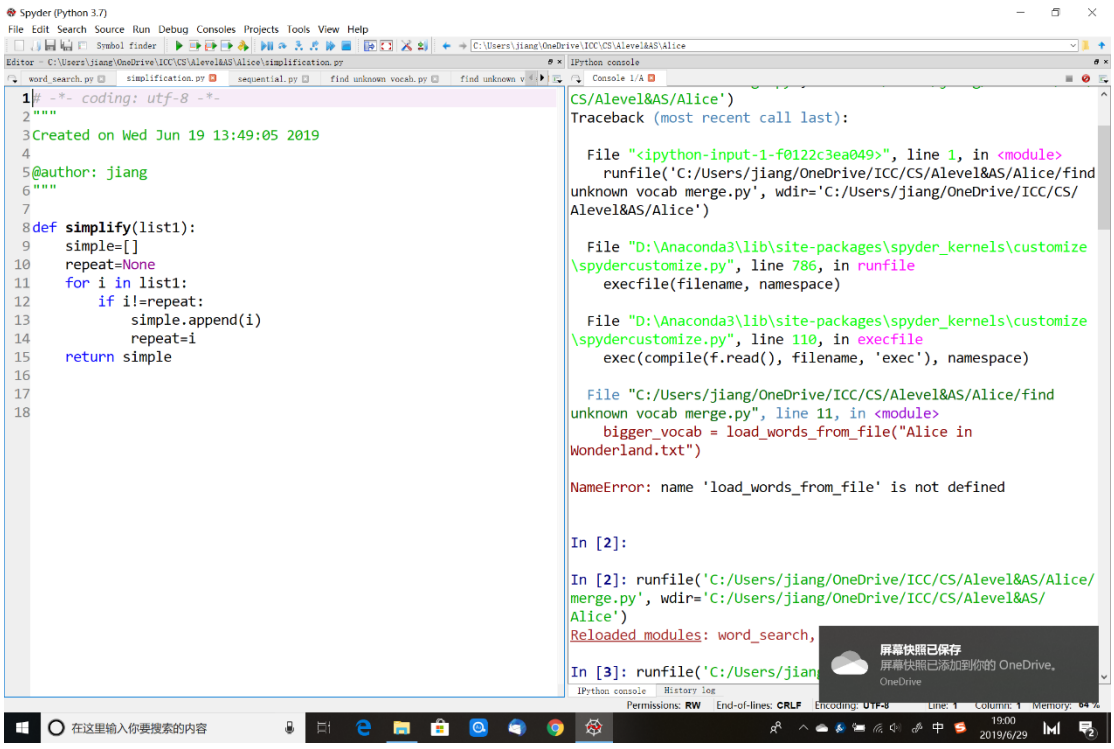
Word search project report

The first step: import the book and the booklet



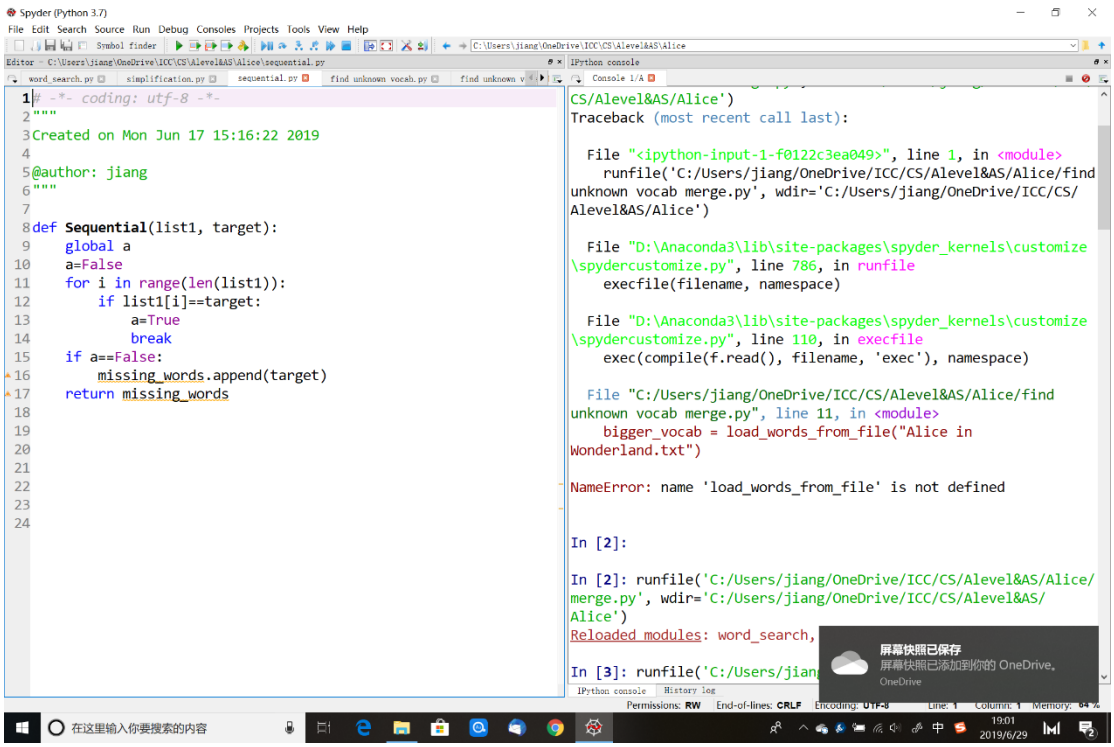
The first function is used to substitute all upper case letters, numbers and notations into spaces. The second function is used to import the book.

The second step: simplify the book

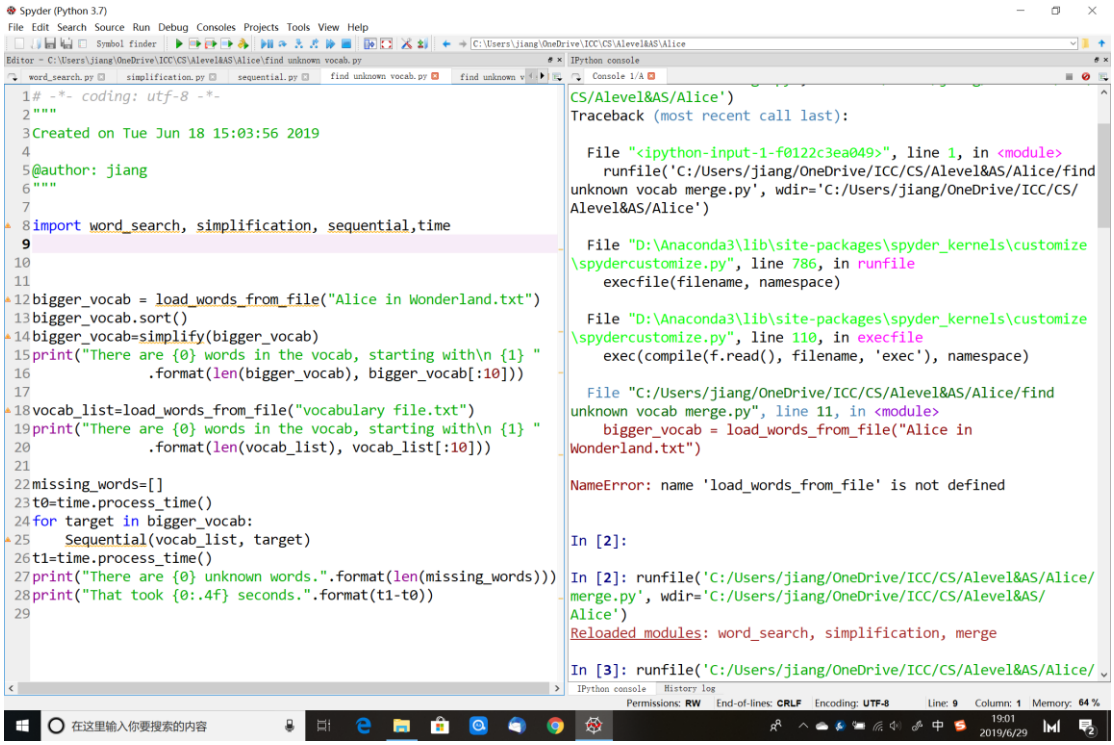


This function is used to remove repeated words in the book.

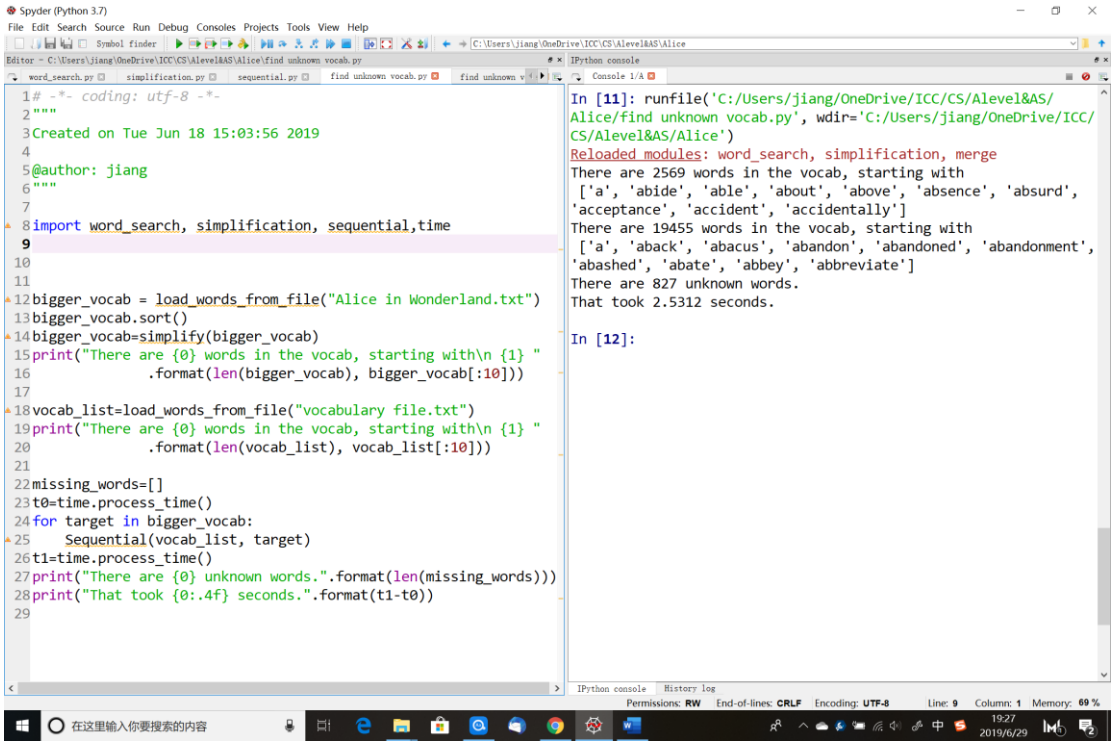
The third step: try the sequential search algorithm



This is the function of sequential search.



This the completed algorithm for sequential search. The first step is to import all the functions needed. After that, I imported the book and the booklet using the word_search function. Then, use the sequential function to find unknown words, at the same time record the time used.

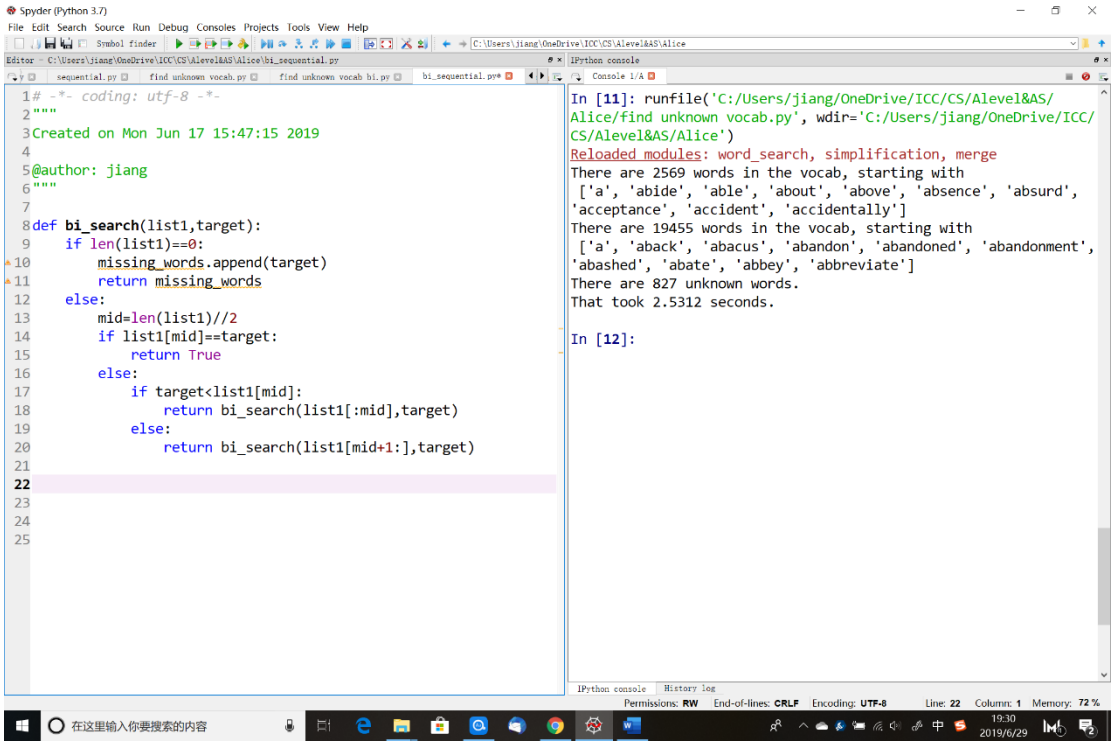


```
# -*- coding: utf-8 -*-
"""
Created on Tue Jun 18 15:03:56 2019

@author: jiang
"""

import word_search, simplification, sequential,time



bigger_vocab = load_words_from_file("Alice in Wonderland.txt")
bigger_vocab.sort()
bigger_vocab=simplify(bigger_vocab)
print("There are {0} words in the vocab, starting with\n {1} "
              .format(len(bigger_vocab), bigger_vocab[:10]))

vocab_list=load_words_from_file("vocabulary file.txt")
print("There are {0} words in the vocab, starting with\n {1} "
              .format(len(vocab_list), vocab_list[:10]))

missing_words=[]
t0=time.process_time()
for target in bigger_vocab:
    Sequential(vocab_list, target)
t1=time.process_time()
print("There are {0} unknown words.".format(len(missing_words)))
print("That took {0:.4f} seconds.".format(t1-t0))
```

```
In [11]: runfile('C:/Users/jiang/OneDrive/ICC/CS/Alevel&AS/
Alice/find unknown vocab.py', wdir='C:/Users/jiang/OneDrive/ICC/
CS/Alevel&AS/Alice')
Reloaded modules: word_search, simplification, merge
There are 2569 words in the vocab, starting with
 ['a', 'abide', 'able', 'about', 'above', 'absence', 'absurd',
'acceptance', 'accident', 'accidentally']
There are 19455 words in the vocab, starting with
 ['a', 'aback', 'abacus', 'abandon', 'abandoned', 'abandonment',
'abashed', 'abate', 'abbey', 'abbreviate']
There are 827 unknown words.
That took 2.5312 seconds.

In [12]:
```

The sequential method used 2.5s to find 827 unknown words in 2569 words from the book.

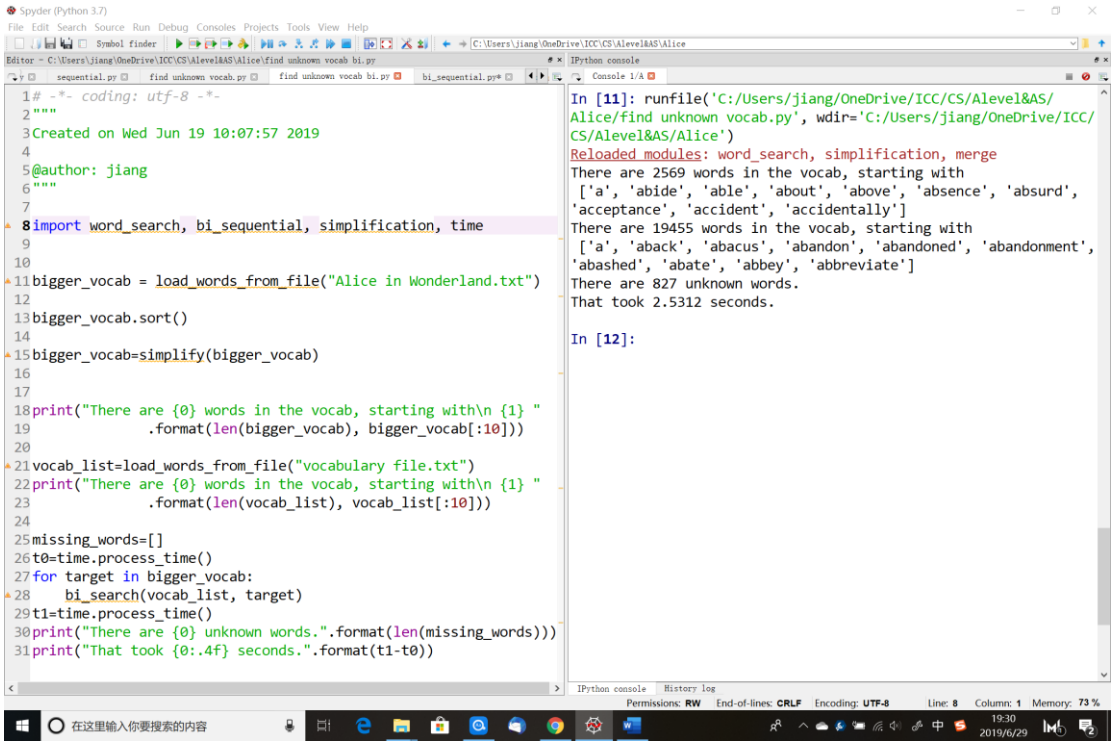
The fourth step: try the binary search algorithm



```
# -*- coding: utf-8 -*-
"""
Created on Mon Jun 17 15:47:15 2019

@author: jiang
"""

def bi_search(list1,target):
    if len(list1)==0:
        missing_words.append(target)
        return missing_words
    else:
        mid=len(list1)//2
        if list1[mid]==target:
            return True
        else:
            if target<list1[mid]:
                return bi_search(list1[:mid],target)
            else:
                return bi_search(list1[mid+1:],target)
```

```
In [11]: runfile('C:/Users/jiang/OneDrive/ICC/CS/Alevel&AS/
Alice/find unknown vocab.py', wdir='C:/Users/jiang/OneDrive/ICC/
CS/Alevel&AS/Alice')
Reloaded modules: word_search, simplification, merge
There are 2569 words in the vocab, starting with
 ['a', 'abide', 'able', 'about', 'above', 'absence', 'absurd',
'acceptance', 'accident', 'accidentally']
There are 19455 words in the vocab, starting with
 ['a', 'aback', 'abacus', 'abandon', 'abandoned', 'abandonment',
'abashed', 'abate', 'abbey', 'abbreviate']
There are 827 unknown words.
That took 2.5312 seconds.

In [12]:
```

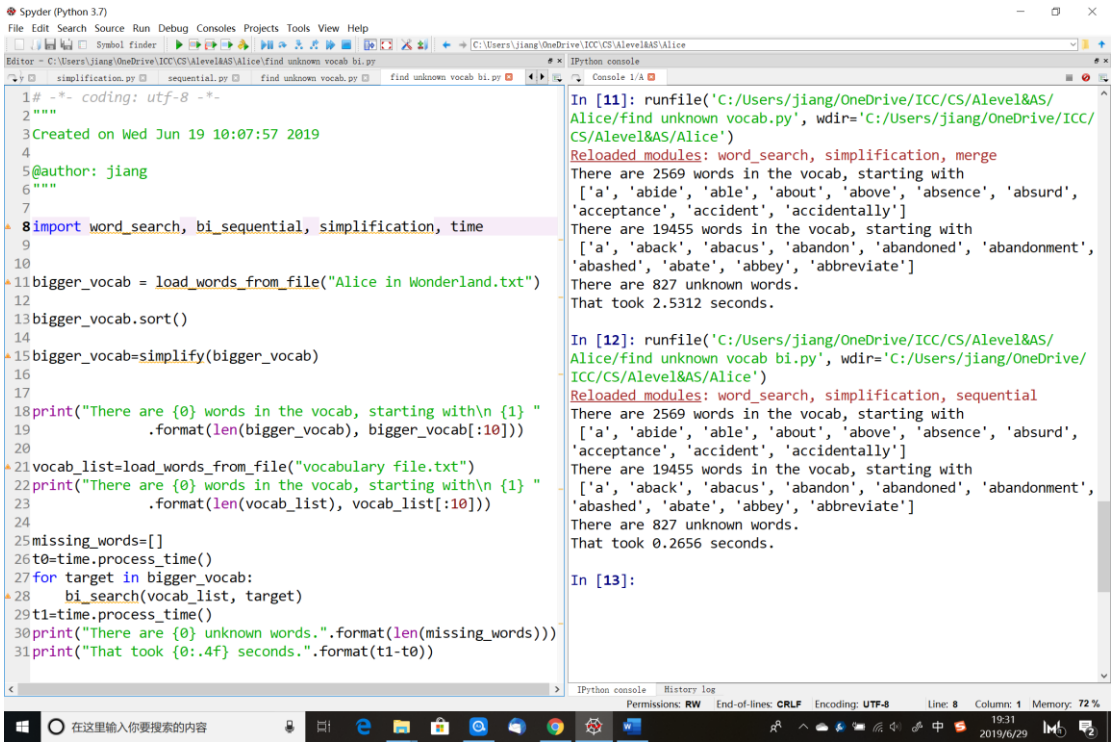
The function of binary search



```
# -*- coding: utf-8 -*-
"""
Created on Wed Jun 19 10:07:57 2019

@author: jiang
"""

import word_search, bi_sequential, simplification, time


bigger_vocab = load_words_from_file("Alice in Wonderland.txt")

bigger_vocab.sort()

bigger_vocab=simplify(bigger_vocab)


print("There are {0} words in the vocab, starting with\n {1} "
              .format(len(bigger_vocab), bigger_vocab[:10]))

vocab_list=load_words_from_file("vocabulary file.txt")
print("There are {0} words in the vocab, starting with\n {1} "
              .format(len(vocab_list), vocab_list[:10]))

missing_words=[]
t0=time.process_time()
for target in bigger_vocab:
    bi_search(vocab_list, target)
t1=time.process_time()
print("There are {0} unknown words.".format(len(missing_words)))
print("That took {0:.4f} seconds.".format(t1-t0))
```

```
In [11]: runfile('C:/Users/jiang/OneDrive/ICC/CS/Alevel&AS/
Alice/find unknown vocab.py', wdir='C:/Users/jiang/OneDrive/ICC/
CS/Alevel&AS/Alice')
Reloaded modules: word_search, simplification, merge
There are 2569 words in the vocab, starting with
 ['a', 'abide', 'able', 'about', 'above', 'absence', 'absurd',
'acceptance', 'accident', 'accidentally']
There are 19455 words in the vocab, starting with
 ['a', 'aback', 'abacus', 'abandon', 'abandoned', 'abandonment',
'abashed', 'abate', 'abbey', 'abbreviate']
There are 827 unknown words.
That took 2.5312 seconds.

In [12]:
```

Do the similar steps as in the sequential algorithm.

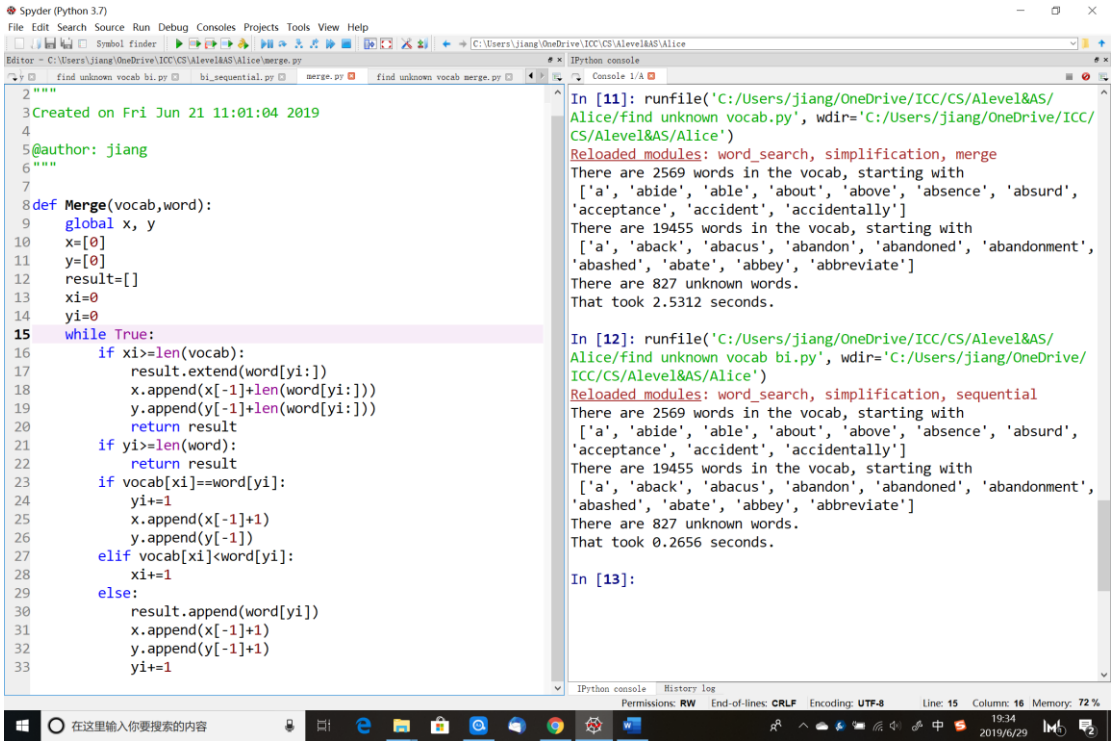


```
# -*- coding: utf-8 -*-
"""
Created on Wed Jun 19 10:07:57 2019

@author: jiang
"""

import word_search, bi_sequential, simplification, time


bigger_vocab = load_words_from_file("Alice in Wonderland.txt")

bigger_vocab.sort()

bigger_vocab=simplify(bigger_vocab)


print("There are {0} words in the vocab, starting with\n {1} "
              .format(len(bigger_vocab), bigger_vocab[:10]))

vocab_list=load_words_from_file("vocabulary file.txt")
print("There are {0} words in the vocab, starting with\n {1} "
              .format(len(vocab_list), vocab_list[:10]))

missing_words=[]
t0=time.process_time()
for target in bigger_vocab:
    bi_search(vocab_list, target)
t1=time.process_time()
print("There are {0} unknown words.".format(len(missing_words)))
print("That took {0:.4f} seconds.".format(t1-t0))
```

```
In [11]: runfile('C:/Users/jiang/OneDrive/ICC/CS/Alevel&AS/
Alice/find unknown vocab.py', wdir='C:/Users/jiang/OneDrive/ICC/
CS/Alevel&AS/Alice')
Reloaded modules: word_search, simplification, merge
There are 2569 words in the vocab, starting with
 ['a', 'abide', 'able', 'about', 'above', 'absence', 'absurd',
'acceptance', 'accident', 'accidentally']
There are 19455 words in the vocab, starting with
 ['a', 'aback', 'abacus', 'abandon', 'abandoned', 'abandonment',
'abashed', 'abate', 'abbey', 'abbreviate']
There are 827 unknown words.
That took 2.5312 seconds.

In [12]: runfile('C:/Users/jiang/OneDrive/ICC/CS/Alevel&AS/
Alice/find unknown vocab bi.py', wdir='C:/Users/jiang/OneDrive/
ICC/CS/Alevel&AS/Alice')
Reloaded modules: word_search, simplification, sequential
There are 2569 words in the vocab, starting with
 ['a', 'abide', 'able', 'about', 'above', 'absence', 'absurd',
'acceptance', 'accident', 'accidentally']
There are 19455 words in the vocab, starting with
 ['a', 'aback', 'abacus', 'abandon', 'abandoned', 'abandonment',
'abashed', 'abate', 'abbey', 'abbreviate']
There are 827 unknown words.
That took 0.2656 seconds.

In [13]:
```

The binary method only use 0.27s to do the same thing, much faster than the sequential one.

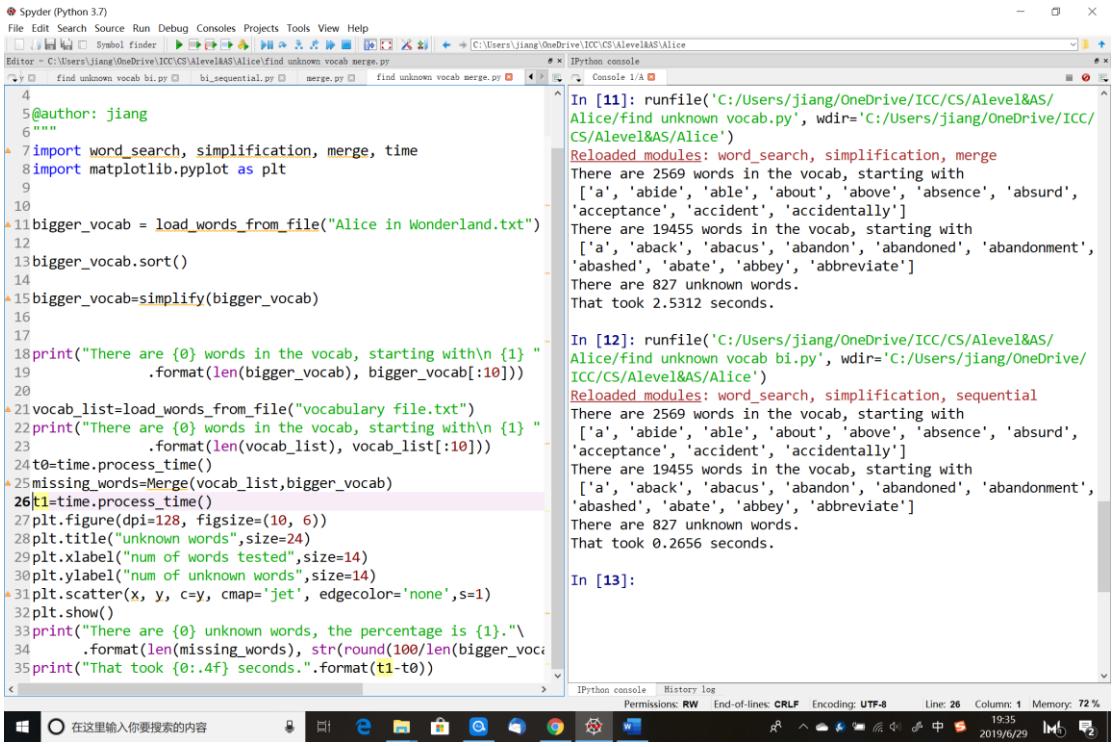
The fifth step: try the merging search algorithm



```
"""
Created on Fri Jun 21 11:01:04 2019

@author: jiang
"""

def Merge(vocab,word):
    global x, y
    x=[0]
    y=[0]
    result=[]
    xi=0
    yi=0
    while True:
        if xi>=len(vocab):
            result.extend(word[yi:])
            x.append(x[-1]+len(word[yi:]))
            y.append(y[-1]+len(word[yi:]))
            return result
        if yi>=len(word):
            return result
        if vocab[xi]==word[yi]:
            yi+=1
            x.append(x[-1]+1)
            y.append(y[-1])
        elif vocab[xi]<word[yi]:
            xi+=1
        else:
            result.append(word[yi])
            x.append(x[-1]+1)
            y.append(y[-1]+1)
            yi+=1
```

```
In [11]: runfile('C:/Users/jiang/OneDrive/ICC/CS/Alevel&AS/
Alice/find unknown vocab.py', wdir='C:/Users/jiang/OneDrive/ICC/
CS/Alevel&AS/Alice')
Reloaded modules: word_search, simplification, merge
There are 2569 words in the vocab, starting with
 ['a', 'abide', 'able', 'about', 'above', 'absence', 'absurd',
'acceptance', 'accident', 'accidentally']
There are 19455 words in the vocab, starting with
 ['a', 'aback', 'abacus', 'abandon', 'abandoned', 'abandonment',
'abashed', 'abate', 'abbey', 'abbreviate']
There are 827 unknown words.
That took 2.5312 seconds.

In [12]: runfile('C:/Users/jiang/OneDrive/ICC/CS/Alevel&AS/
Alice/find unknown vocab bi.py', wdir='C:/Users/jiang/OneDrive/
ICC/CS/Alevel&AS/Alice')
Reloaded modules: word_search, simplification, sequential
There are 2569 words in the vocab, starting with
 ['a', 'abide', 'able', 'about', 'above', 'absence', 'absurd',
'acceptance', 'accident', 'accidentally']
There are 19455 words in the vocab, starting with
 ['a', 'aback', 'abacus', 'abandon', 'abandoned', 'abandonment',
'abashed', 'abate', 'abbey', 'abbreviate']
There are 827 unknown words.
That took 0.2656 seconds.

In [13]:
```

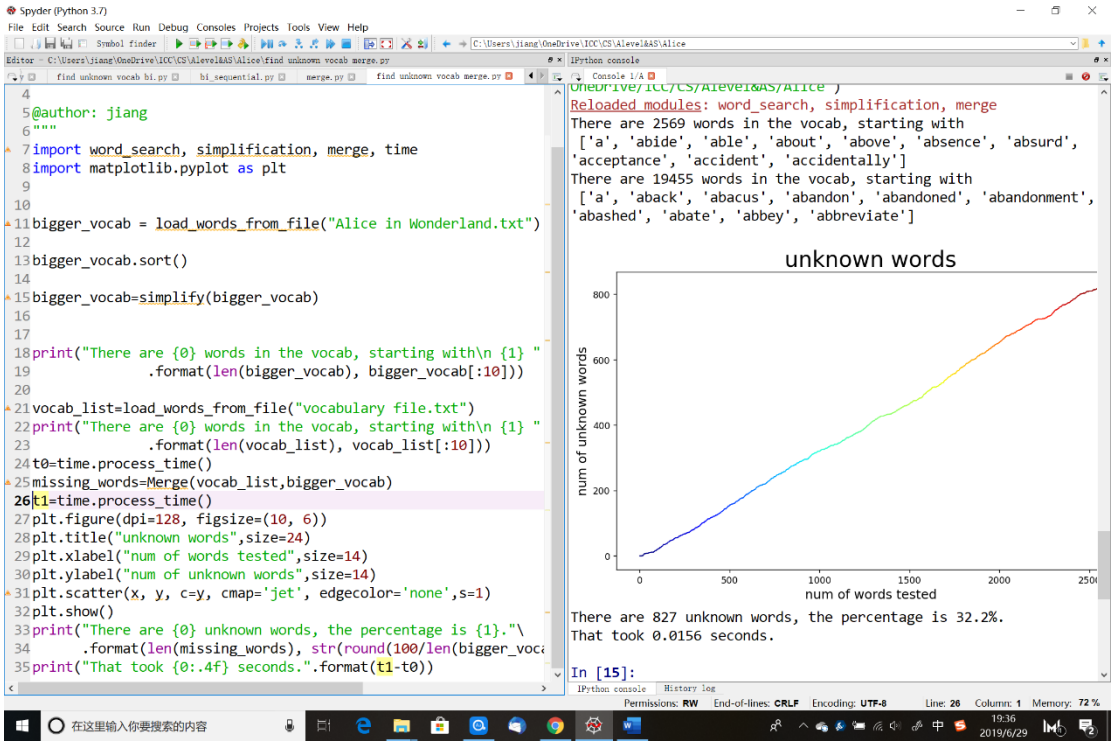
The function of merging search



```
@author: jiang
"""
import word_search, simplification, merge, time
import matplotlib.pyplot as plt


bigger_vocab = load_words_from_file("Alice in Wonderland.txt")

bigger_vocab.sort()

bigger_vocab=simplify(bigger_vocab)


print("There are {0} words in the vocab, starting with\n {1} "
              .format(len(bigger_vocab), bigger_vocab[:10]))

vocab_list=load_words_from_file("vocabulary file.txt")
print("There are {0} words in the vocab, starting with\n {1} "
              .format(len(vocab_list), vocab_list[:10]))
t0=time.process_time()
missing_words=Merge(vocab_list,bigger_vocab)
t1=time.process_time()
plt.figure(dpi=128, figsize=(10, 6))
plt.title("unknown words",size=24)
plt.xlabel("num of words tested",size=14)
plt.ylabel("num of unknown words",size=14)
plt.scatter(x, y, c=y, cmap='jet', edgecolor='none',s=1)
plt.show()
print("There are {0} unknown words, the percentage is {1}."\
      .format(len(missing_words), str(round(100/len(bigger_voca
print("That took {0:.4f} seconds.".format(t1-t0))
```

```
In [11]: runfile('C:/Users/jiang/OneDrive/ICC/CS/Alevel&AS/
Alice/find unknown vocab.py', wdir='C:/Users/jiang/OneDrive/ICC/
CS/Alevel&AS/Alice')
Reloaded modules: word_search, simplification, merge
There are 2569 words in the vocab, starting with
 ['a', 'abide', 'able', 'about', 'above', 'absence', 'absurd',
'acceptance', 'accident', 'accidentally']
There are 19455 words in the vocab, starting with
 ['a', 'aback', 'abacus', 'abandon', 'abandoned', 'abandonment',
'abashed', 'abate', 'abbey', 'abbreviate']
There are 827 unknown words.
That took 2.5312 seconds.

In [12]: runfile('C:/Users/jiang/OneDrive/ICC/CS/Alevel&AS/
Alice/find unknown vocab bi.py', wdir='C:/Users/jiang/OneDrive/
ICC/CS/Alevel&AS/Alice')
Reloaded modules: word_search, simplification, sequential
There are 2569 words in the vocab, starting with
 ['a', 'abide', 'able', 'about', 'above', 'absence', 'absurd',
'acceptance', 'accident', 'accidentally']
There are 19455 words in the vocab, starting with
 ['a', 'aback', 'abacus', 'abandon', 'abandoned', 'abandonment',
'abashed', 'abate', 'abbey', 'abbreviate']
There are 827 unknown words.
That took 0.2656 seconds.

In [13]:
```

Do about the same thing as before, but this time I calculated the percentage of unknown words and plotted a graph of the searching process.



This time, it only takes 0.0156s to finish the searching.