

1402-02-08



Persian Gulf University

Faculty of Intelligent Systems Engineering and Data Science

Natural Language Processing

Dr. Mohammad Bidoki

Homework #2

By: Mohammad Barzegar

Student ID: 4010724001

تمرین شماره 1 (WhiteSpaceTokenizer):

با استفاده از متد `WhiteSpaceTokenizer` کاری میکنیم تا فاصله ها (اسپیس یا تب) و پاراگراف ها را حذف کرده و بین هر کلمه فقط یک اسپیس وجود داشته باشد.

```

1 import nltk
2 from nltk.tokenize import WhitespaceTokenizer

1 # QUESTION 1 - White Space
2
3 # Define input and output filenames, and Encodings for each file
4 input_filenames = ["zahak.txt", "ShortSamplePersian.txt", "ShortSampleEnglish.txt", "Beanstalk.txt"]
5 output_filenames = ["zahak1.txt", "ShortSamplePersian1.txt", "ShortSampleEnglish1.txt", "Beanstalk1.txt"]
6 encoding_filenames = ["utf-8-sig", "utf-8-sig", "utf-8", "ANSI"]
7
8
9 # Loop over each input file and tokenize the text
10 for i in range(len(input_filenames)):
11     # Load input text file
12     with open(input_filenames[i], "r", encoding=encoding_filenames[i]) as f:
13         input_text = f.read()
14
15     # Tokenize the input text using WhiteSpaceTokenizer
16     tokenizer = WhitespaceTokenizer()
17     tokens = tokenizer.tokenize(input_text)
18     output_text = ' '.join(tokens)
19
20     # Save the tokens to an output text file
21     with open(output_filenames[i], "w", encoding="utf-8") as f:
22         f.write(output_text)

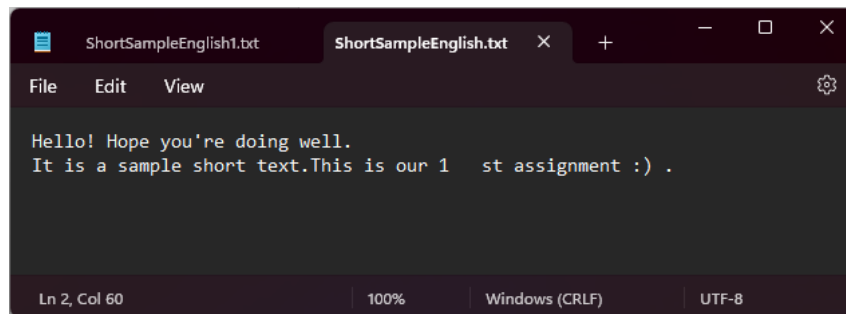
```

پس از ایمپورت کردن کتابخانه های مورد استفاده، اسم فایل های ورودی را در `input_filenames` و اسم مورد نظر برای فایل های خروجی را در لیست `output_filenames` نوشته ایم. انکودینگ هر فایل فرق دارد که در `encoding_filenames` نام انکودینگ هر فایل نوشته شده است (ایندکس مشابه `input_filenames`).

سپس در یک حلقه، تک تک هر فایل ورودی را لود کرده و متد `WhiteSpaceTokenizer` را روی آن انجام میدهیم و سپس با استفاده از دستور `join` کلمات را طوری به هم اتصال میدهیم که فقط یک اسپیس بین آن ها باشد. سپس فایل را ذخیره می کنیم.

برای هر تمرین، ورودی و خروجی کد را برای فایل `ShortSampleEnglish` می بینیم و سایر فایل ها در فولدر تمرین موجود می باشد.

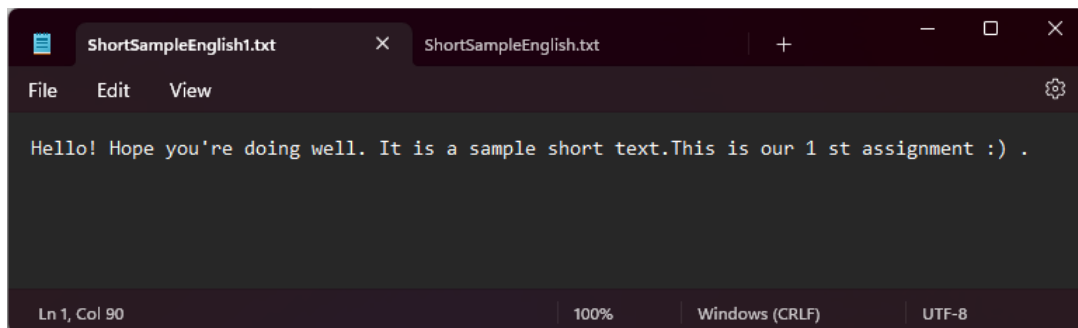
فایل `ShortSampleEnglish` قبل از اعمال این کد:



```

ShortSampleEnglish1.txt  ShortSampleEnglish.txt
File Edit View
Hello! Hope you're doing well.
It is a sample short text.This is our 1  st assignment :) .
Ln 2, Col 60  100%  Windows (CRLF)  UTF-8
  
```

فایل `ShortSampleEnglish` پس از اعمال این کد:



```

ShortSampleEnglish1.txt  ShortSampleEnglish.txt
File Edit View
Hello! Hope you're doing well. It is a sample short text.This is our 1 st assignment :) .
Ln 1, Col 90  100%  Windows (CRLF)  UTF-8
  
```

دقت فرمایید فایل خروجی برای هر سوال، عدد آن سوال را در آخر اسم فایل خود دارد، مثلا `zahak1`

خروجی فایل `zahak` پس از انجام دادن دستورات تمرین 1 است.

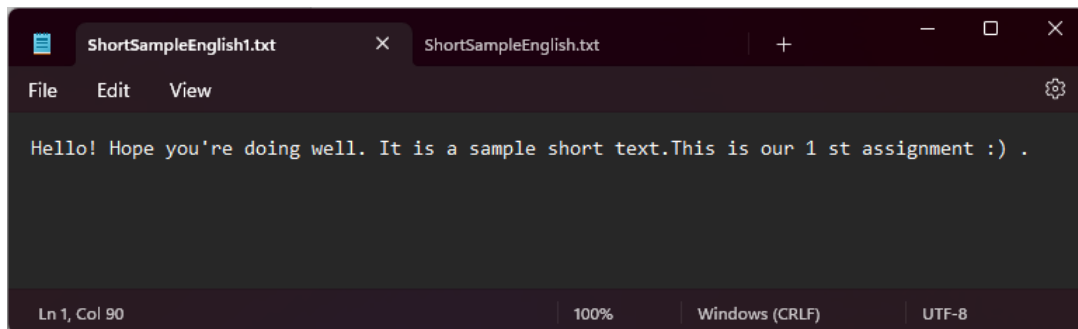
تمرین شماره 2 (Lowercase):

در این سوال، خروجی سوال قبلی را با استفاده از متد `lower` تمامی حروف را به حروف کوچک انگلیسی تبدیل می کنیم. بدیهی است که این کار برای فایل های فارسی نیاز نیست.

روال کلی مشابه سوال قبلی است:

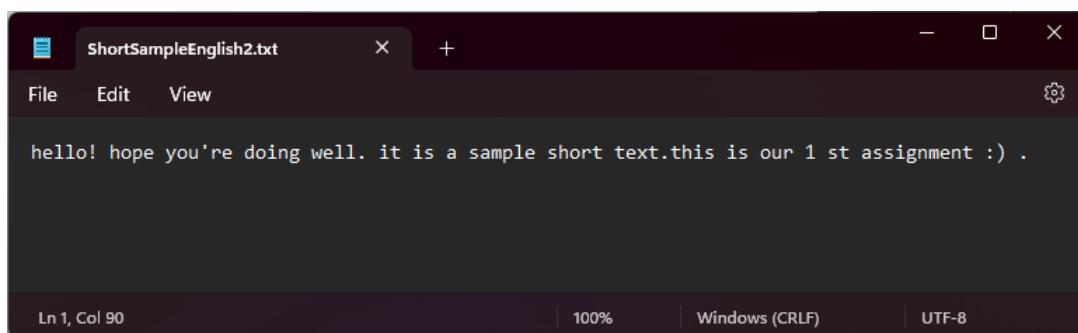
```
1 # QUESTION 2 - making the letters lowercase
2
3 # Define input and output filenames
4 input_filenames = ["ShortSampleEnglish1.txt", "Beanstalk1.txt"]
5 output_filenames = ["ShortSampleEnglish2.txt", "Beanstalk2.txt"]
6
7 # Loop over each input file and lowercase the text
8 for i in range(len(input_filenames)):
9     # Open the input files
10    with open(input_filenames[i], "r", encoding='utf-8') as f:
11        # Read the contents of the files
12        text = f.read()
13
14    # Convert the contents of the files to lowercase
15    text_lower = text.lower()
16
17    # Save the lowercase contents to output files
18    with open(output_filenames[i], "w", encoding='utf-8') as f:
19        f.write(text_lower)
```

فایل ShortSampleEnglish قبل از اعمال این کد:



The screenshot shows a text editor window with two tabs: 'ShortSampleEnglish1.txt' and 'ShortSampleEnglish.txt'. The 'ShortSampleEnglish1.txt' tab is active. The text inside the editor is 'Hello! Hope you're doing well. It is a sample short text.This is our 1 st assignment :) .'. The status bar at the bottom indicates 'Ln 1, Col 90', '100%', 'Windows (CRLF)', and 'UTF-8'.

فایل ShortSampleEnglish پس از اعمال این کد:



The screenshot shows a text editor window with two tabs: 'ShortSampleEnglish2.txt' and '+'. The 'ShortSampleEnglish2.txt' tab is active. The text inside the editor is 'hello! hope you're doing well. it is a sample short text.this is our 1 st assignment :) .'. The status bar at the bottom indicates 'Ln 1, Col 90', '100%', 'Windows (CRLF)', and 'UTF-8'.

تمرین شماره 3 (استخراج جملات و توکن ها):

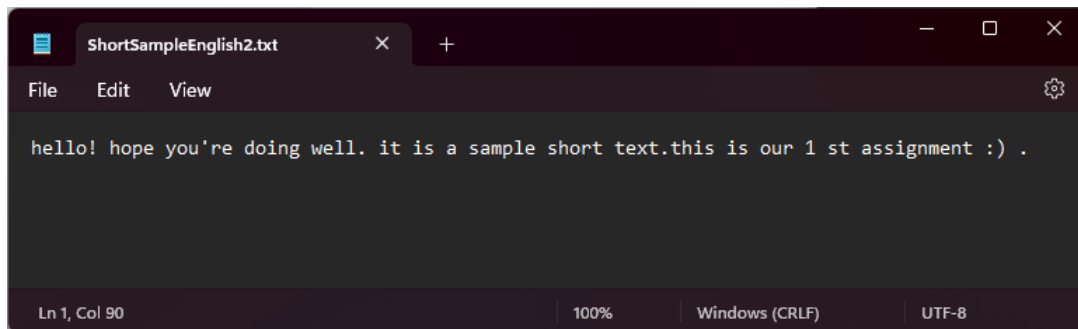
با استفاده از PunktSentenceTokenizer، جملات هر فایل را شناسایی و جداسازی انجام می دهیم. در انتها در هر خط یک جمله قرار خواهد داشت. ورودی متون انگلیسی از تمرین دوم و متون فارسی از تمرین اول است.

```

1 # QUESTION 3 - Tokenize
2 nltk.download('punkt')
3 from nltk.tokenize import PunktSentenceTokenizer
4
5 # Create a list of input file names
6 input_files = ["zahak1.txt", "ShortSamplePersian1.txt", "ShortSampleEnglish2.txt", "Beastalk2.txt"]
7 output_files = ["zahak3.txt", "ShortSamplePersian3.txt", "ShortSampleEnglish3.txt", "Beastalk3.txt"]
8 encoding_filenames = ["utf-8-sig", "utf-8-sig", "utf-8", "ANSI"]
9
10 # Instantiate the PunktSentenceTokenizer
11 tokenizer = PunktSentenceTokenizer()
12
13 # Loop over the input files and tokenize each file
14 for i, file_name in enumerate(input_files):
15     # Open the input file
16     with open(file_name, "r", encoding=encoding_filenames[i]) as f:
17         # Read the contents of the file
18         text = f.read()
19
20         # Tokenize the text into sentences
21         sentences = tokenizer.tokenize(text)
22
23         # Write the sentences to the output file
24         with open(output_files[i], "w", encoding="utf-8") as f_out:
25             f_out.write("\n".join(sentences))

```

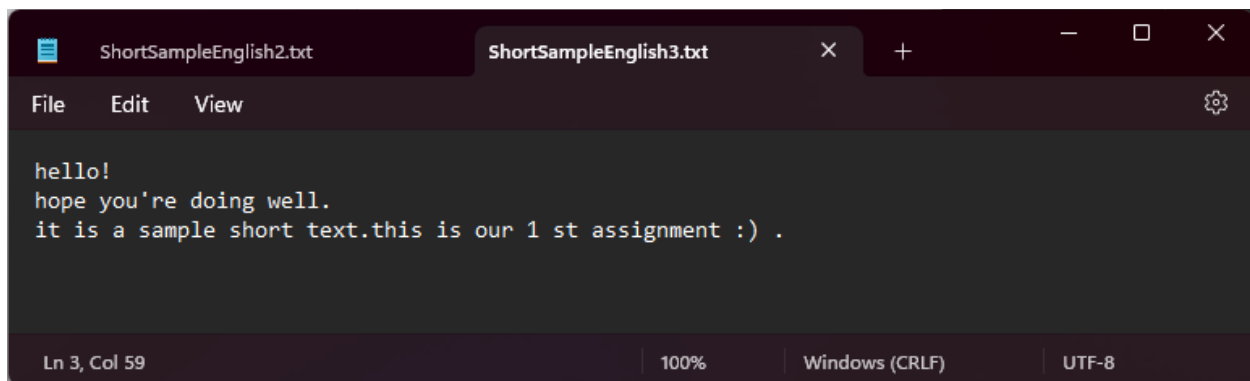
فایل ShortSampleEnglish قبل از اعمال این کد:



A screenshot of a text editor window titled "ShortSampleEnglish2.txt". The editor has a menu bar with "File", "Edit", and "View". The text area contains a single line: "hello! hope you're doing well. it is a sample short text.this is our 1 st assignment :) .". The status bar at the bottom shows "Ln 1, Col 90", "100%", "Windows (CRLF)", and "UTF-8".

```
hello! hope you're doing well. it is a sample short text.this is our 1 st assignment :) .
```

فایل ShortSampleEnglish بعد از اعمال این کد:



A screenshot of a text editor window titled "ShortSampleEnglish3.txt". The editor has a menu bar with "File", "Edit", and "View". The text area contains three lines: "hello!", "hope you're doing well.", and "it is a sample short text.this is our 1 st assignment :) .". The status bar at the bottom shows "Ln 3, Col 59", "100%", "Windows (CRLF)", and "UTF-8".

```
hello!  
hope you're doing well.  
it is a sample short text.this is our 1 st assignment :) .
```

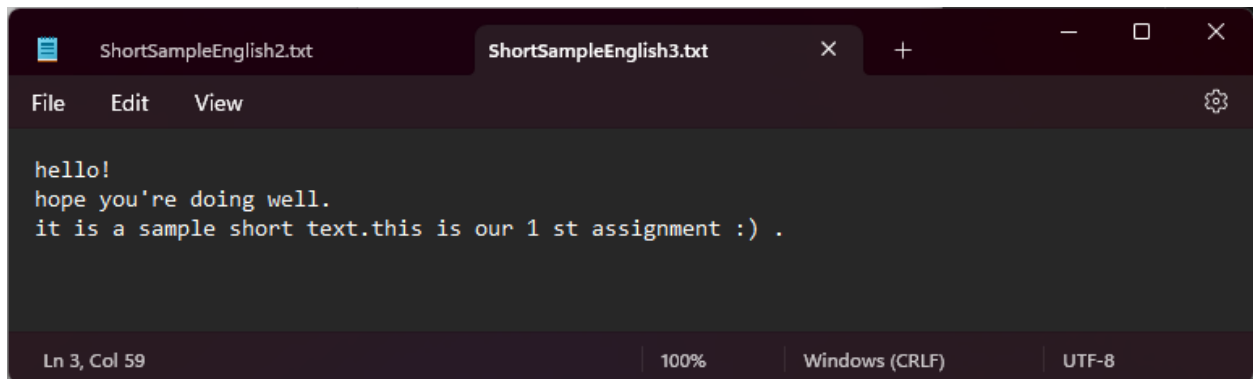
تمرین شماره 4 (حذف علائم نگارشی):

ورودی این مرحله، همان متون خروجی مرحله قبل است. اکنون میخواهیم همه علائم بجز حروف الفبا از متون حذف شود. با استفاده از `RegexpTokenizer` این کار را انجام می دهیم:

```
1 # QUESTION 4 - RegexpTokenizer
2
3 from nltk.tokenize import RegexpTokenizer
4
5 input_files = ["zahak3.txt", "ShortSamplePersian3.txt", "ShortSampleEnglish3.txt", "Beanstalk3.txt"]
6 output_files = ["zahak4.txt", "ShortSamplePersian4.txt", "ShortSampleEnglish4.txt", "Beanstalk4.txt"]
7
8 # Only include Persian and English alphabet letters
9 tokenizer = RegexpTokenizer(r'[ا-زآ-ژسشظطعفتکگلنوهی\u0600-\u06FFa-zA-Z]+')
10
11
12 for i in range(len(input_files)):
13
14     # Create a tokenizer using a regular expression pattern to match words
15
16     # Open the input file
17     with open(input_files[i], 'r', encoding='utf-8') as f:
18         # Read the contents of the file
19         text = f.read()
20
21         # Tokenize the text into words using the tokenizer
22         words = tokenizer.tokenize(text.lower())
23
24         # Write the words to the output file
25         with open(output_files[i], "w", encoding="utf-8") as f_out:
26             f_out.write(" ".join(words))
27
```

در لاین 9 به عنوان argument دستور `RegexpTokenizer` حروف الفبای فارسی و انگلیسی را قرار می دهیم، تا همه ی کاراکتر ها بجز 34 حرف فارسی و حروف بزرگ و کوچک انگلیسی `a-z` از متن حذف شود.

فایل ShortSampleEnglish قبل از اعمال این کد:

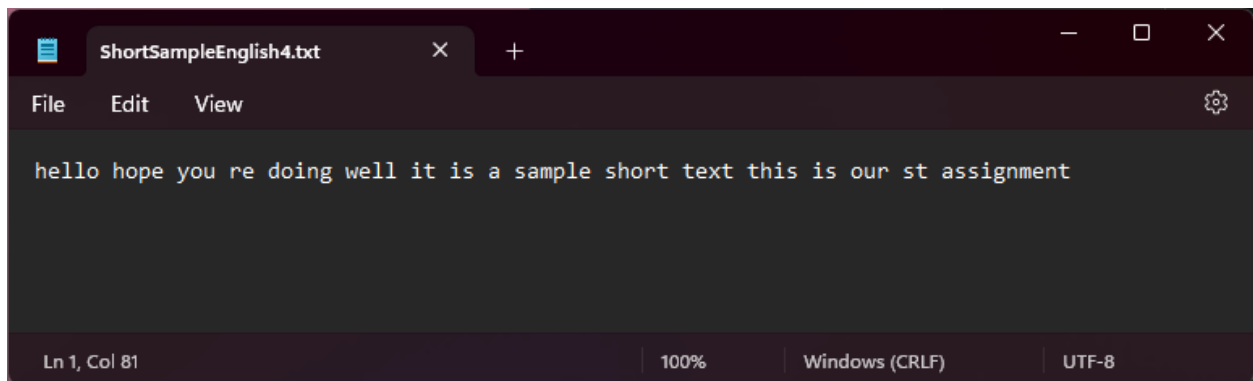


The screenshot shows a text editor window with two tabs: 'ShortSampleEnglish2.txt' and 'ShortSampleEnglish3.txt'. The 'ShortSampleEnglish3.txt' tab is active. The editor has a menu bar with 'File', 'Edit', and 'View'. The text content is as follows:

```
hello!  
hope you're doing well.  
it is a sample short text.this is our 1 st assignment :) .
```

The status bar at the bottom indicates 'Ln 3, Col 59', '100%', 'Windows (CRLF)', and 'UTF-8'.

فایل ShortSampleEnglish قبل از اعمال این کد:



The screenshot shows a text editor window with one tab: 'ShortSampleEnglish4.txt'. The editor has a menu bar with 'File', 'Edit', and 'View'. The text content is as follows:

```
hello hope you re doing well it is a sample short text this is our st assignment
```

The status bar at the bottom indicates 'Ln 1, Col 81', '100%', 'Windows (CRLF)', and 'UTF-8'.

تمرین شماره 5 (StopWords):

Stop Words اصطلاحاً کلماتی هستند که در یک زبان با تکرار بسیار زیاد استفاده می شوند ولی معنای خاصی

به متن ما اضافه نمی کنند. مثلاً در زبان فارسی (آنچه، آنگاه، آن، آهان) و در انگلیسی (how, by, just).

لیست این کلمات انگلیسی در کتابخانه nltk موجود است و آن را ایمپورت میکنیم. ابتدا برای متن های انگلیسی

این کار را انجام می دهیم:

```

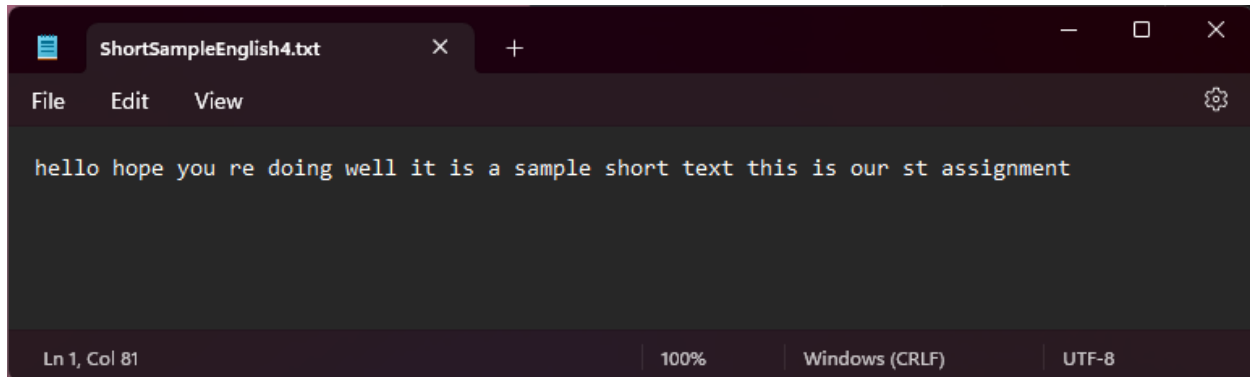
1 # QUESTION 5 - StopWord - English texts
2 nltk.download('stopwords')
3
4 from nltk.corpus import stopwords
5 from nltk.tokenize import word_tokenize
6
7 input_files = ["ShortSampleEnglish4.txt", "Beanstalk4.txt"]
8 output_files = ["ShortSampleEnglish5.txt", "Beanstalk5.txt"]
9
10 # Load the stopwords
11 stop_words = set(stopwords.words('english'))
12
13 for i in range(len(input_files)):
14     # Open the input file
15     with open(input_files[i], 'r', encoding='utf-8') as f:
16         # Read the contents of the file
17         text = f.read()
18
19         words = word_tokenize(text)
20
21         # Remove the stopwords from the list of words
22         cleaned_words = [word for word in words if word not in stop_words]
23
24
25         # Join the cleaned words into a string with spaces between them
26         cleaned_text = ' '.join(cleaned_words)
27
28         # Write the cleaned text to an output file
29         with open(output_files[i], 'w', encoding='utf-8') as f_out:
30             f_out.write(cleaned_text)
31     print(input_files[i], 'Before:', len(text), 'After:', len(cleaned_text))

```

ShortSampleEnglish4.txt Before: 80 After: 47
Beanstalk4.txt Before: 4265 After: 2635

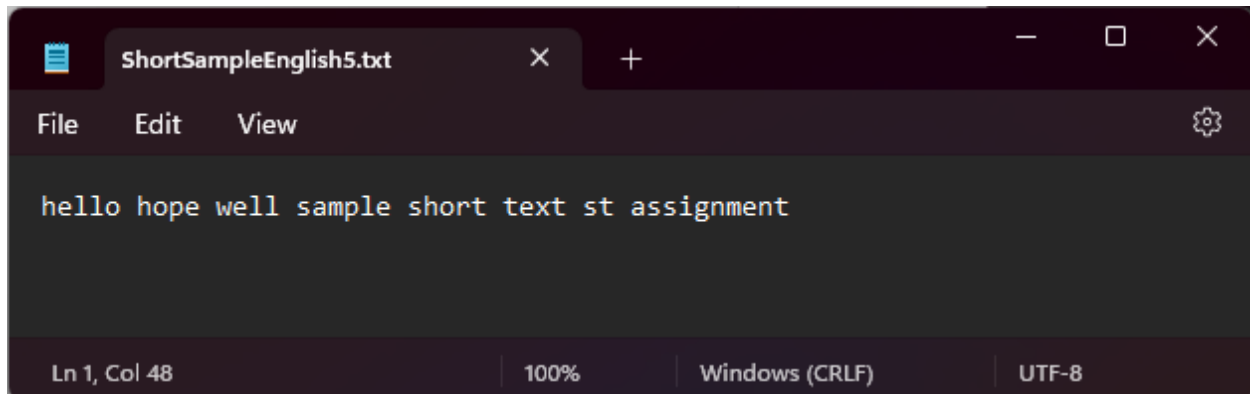
مشاهده می شود که یک فایل از 80 کاراکتر به 47 کاراکتر کاهش یافته، و فایل دیگر از 4265 به 2635 کاهش یافته است.

فایل ShortSampleEnglish قبل از اعمال این کد:



The screenshot shows a text editor window titled 'ShortSampleEnglish4.txt'. The menu bar includes 'File', 'Edit', and 'View'. The text area contains the sentence: 'hello hope you re doing well it is a sample short text this is our st assignment'. The status bar at the bottom indicates 'Ln 1, Col 81', '100%', 'Windows (CRLF)', and 'UTF-8'.

فایل ShortSampleEnglish بعد از اعمال این کد:



The screenshot shows a text editor window titled 'ShortSampleEnglish5.txt'. The menu bar includes 'File', 'Edit', and 'View'. The text area contains the modified sentence: 'hello hope well sample short text st assignment'. The status bar at the bottom indicates 'Ln 1, Col 48', '100%', 'Windows (CRLF)', and 'UTF-8'.

برای زبان فارسی در این کتابخانه **stopword** ها وجود ندارد و باید از منبعی دیگری آن ها را پیدا کنیم. یک برنامه نویس ایرانی به نام **vahid kharazi** لیستی از **stopword** های فارسی تهیه کرده و در گیتهاب قرار داده است، این لیست کلمات را استخراج و در فایل تکست ذخیره می کنیم (**persian_stopwords.txt**).

لینک گیتهاب: <https://github.com/kharazi/persian-stopwords>

```

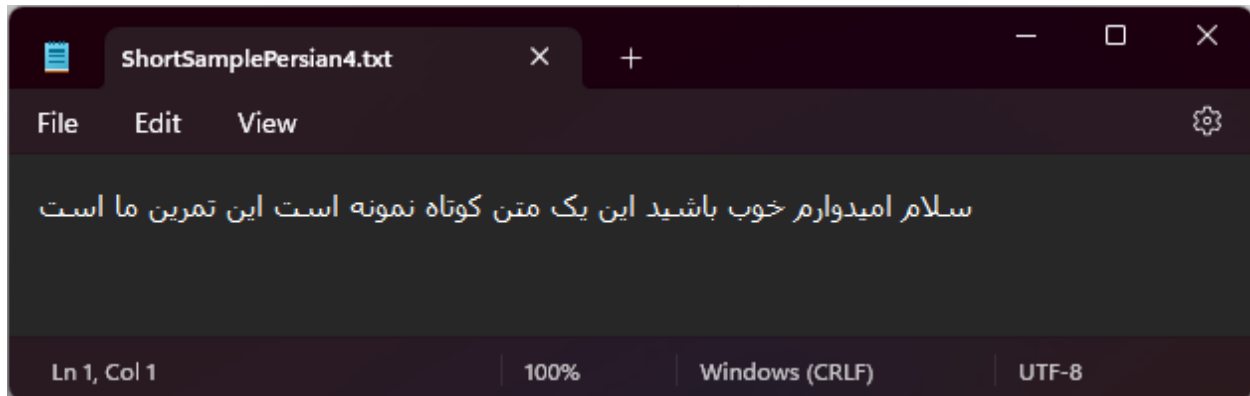
1 # QUESTION 5 - StopWord - Persian texts
2
3 # Open the file containing Persian stopwords to be removed
4 with open('persian_stopwords.txt', 'r', encoding='utf-8') as f_remove:
5     # Read the contents of the file and split them into a list of words
6     words_to_remove = f_remove.read().split()
7
8 input_files = ["zahak4.txt", "ShortSamplePersian4.txt"]
9 output_files = ["zahak5.txt", "ShortSamplePersian5.txt"]
10
11 for i in range(len(input_files)):
12     # Open the input file
13     with open(input_files[i], 'r', encoding='utf-8') as f:
14         # Read the contents of the file
15         text = f.read()
16
17         # Split the text into a list of words
18         words = text.split()
19
20         # Remove the words from the list of words
21         cleaned_words = [word for word in words if word not in words_to_remove]
22
23         # Join the cleaned words into a string with spaces between them
24         cleaned_text = ' '.join(cleaned_words)
25
26         # Write the cleaned text to an output file
27         with open(output_files[i], 'w', encoding='utf-8') as f_out:
28             f_out.write(cleaned_text)
29
30     print(input_files[i], 'Before:', len(text), 'After:', len(cleaned_text))

```

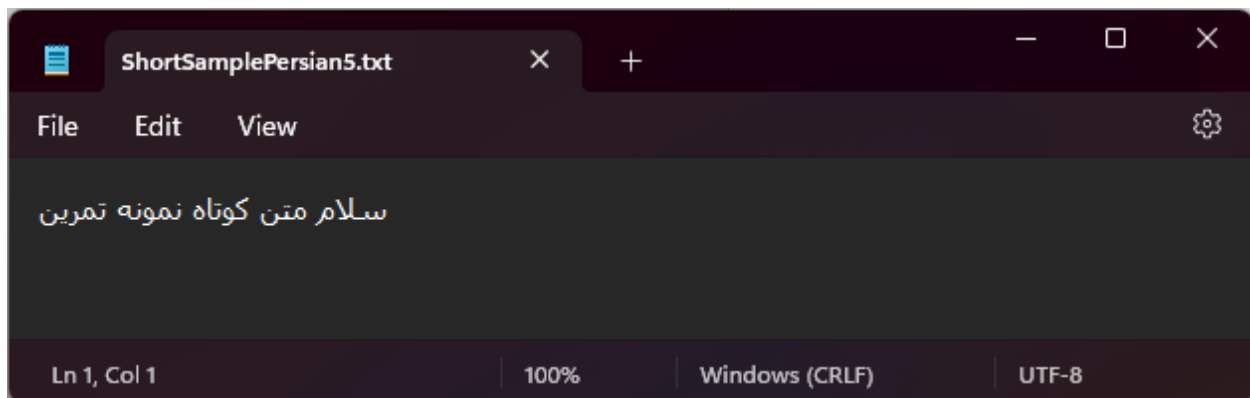
zahak4.txt Before: 5506 After: 3495
 ShortSamplePersian4.txt Before: 67 After: 26

مشاهده می شود که یک متن از 5506 به 3495 کاراکتر و متن دیگر از 67 به 26 کاراکتر کاهش یافته است. این کار برای کاهش حجم فایل ها و همچنین حذف کلامتی که معنای خاصی به متن اضافه نمی کنند موثر است.

فایل ShortSamplePersian قبل از اعمال این کد:



فایل ShortSamplePersian پس از اعمال این کد:



تمرین شماره 6 (Stemming):

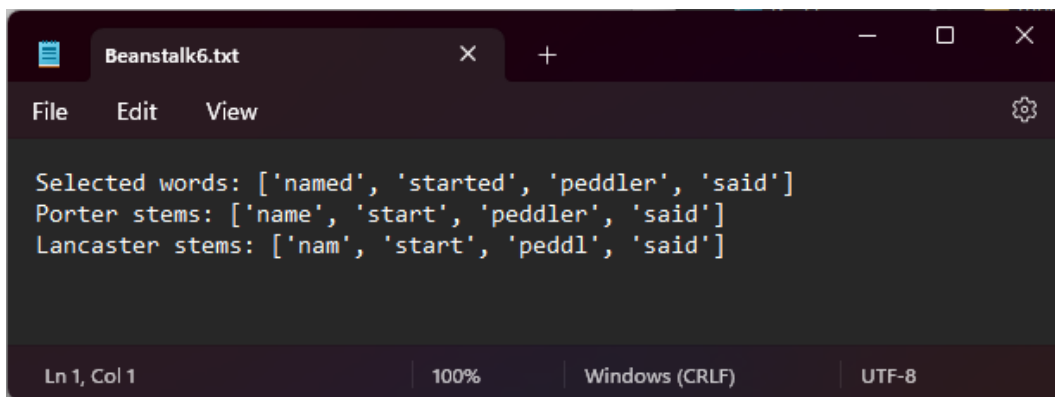
در این قسمت با PorterStemmer و LancasterStemmer ریشه کلمات گفته شده در ایندکس های مورد نظر را استخراج خواهیم کرد.

```

1 #Question 6 - Stemming
2
3 from nltk.stem import PorterStemmer, LancasterStemmer
4 from nltk.tokenize import word_tokenize
5
6 input_files = ["ShortSampleEnglish5.txt", "Beanstalk5.txt"]
7 output_files = ["ShortSampleEnglish6.txt", "Beanstalk6.txt"]
8 indices1 = [2]
9 indices2 = [3, 11, 60, 68]
10 indices = [indices1, indices2]
11
12 # Create stemmers
13 porter = PorterStemmer()
14 lancaster = LancasterStemmer()
15
16 for i in range(len(input_files)):
17     # Read input text file
18     with open(input_files[i], 'r', encoding='utf-8') as file:
19         text = file.read()
20
21     # Tokenize the text into words
22     words = word_tokenize(text)
23
24     # Select words at specific indices
25     selected_words = [words[i] for i in indices[i]]
26
27     # Stem the words using PorterStemmer and LancasterStemmer
28     porter_stems = [porter.stem(word) for word in selected_words]
29     lancaster_stems = [lancaster.stem(word) for word in selected_words]
30
31     # Save the stems to output file
32     with open(output_files[i], 'w', encoding='utf-8') as file:
33         file.write(f"Selected words: {selected_words}\n")
34         file.write(f"Porter stems: {porter_stems}\n")
35         file.write(f"Lancaster stems: {lancaster_stems}")

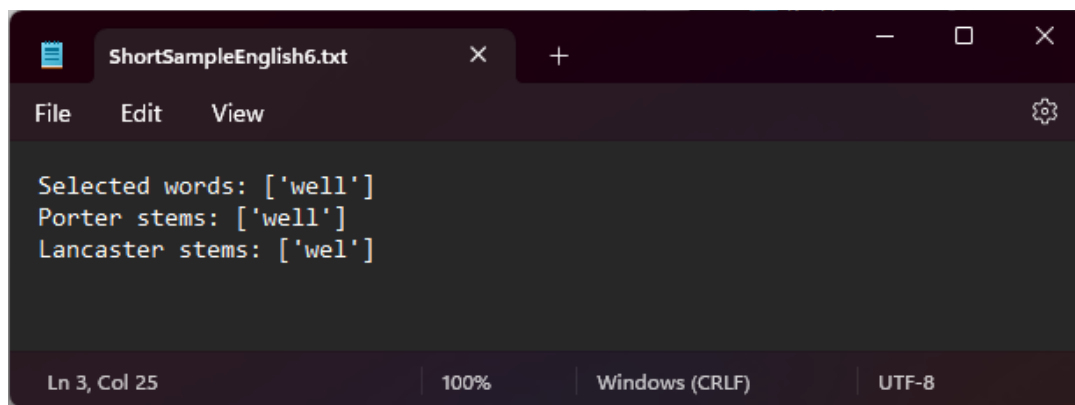
```

خروجی برای BeansTalk:



```
Selected words: ['named', 'started', 'peddler', 'said']
Porter stems: ['name', 'start', 'peddler', 'said']
Lancaster stems: ['nam', 'start', 'peddl', 'said']
```

خروجی برای ShortSampleEnglish:



```
Selected words: ['well']
Porter stems: ['well']
Lancaster stems: ['wel']
```

در سطر اول کلمات اصلی، در سطر دوم خروجی porter و در سطر سوم خروجی Lancaster نوشته شده است.

تمرین شماره 7 (Lemmatization):

با استفاده از WordNetLemmatizer کلمات را به حالت نگارشی اولیه بر میگردانیم.

ابتدا با ورودی های پیشفرض متد lemmatize متوجه می شویم که پاسخ درست برای همه ی کلمات بر

نمیگردد :

```
went -> went
better -> better
was -> wa
eaten -> eaten
butterflies -> butterfly
fishing -> fishing
signaling -> signaling
```

سپس با تعریف کردن آرگومان pos در متد lemmatize، برای هر کلمه تعریف می کنیم که این کلمه

فعل، اسم، صفت یا چه نوع کلمه دیگری است، سپس مشاهده شد که خروجی ها درست تولید می شود:

```
1 #Question 7 - Stemming
2
3 from nltk.stem import WordNetLemmatizer
4 nltk.download('omw-1.4')
5
6
7 lemmatizer = WordNetLemmatizer()
8
9 ADJ, ADJ_SAT, ADV, NOUN, VERB = 'a', 's', 'r', 'n', 'v'
10 words = ['went', 'better', 'was', 'eaten', 'butterflies', 'fishing', 'signaling']
11 pos = [VERB, ADJ, VERB, VERB, NOUN, VERB, VERB]
12
13 for i, word in enumerate(words):
14     lemma = lemmatizer.lemmatize(word, pos=pos[i])
15     print(f"{word} -> {lemma}")
16
```



```
went -> go
better -> good
was -> be
eaten -> eat
butterflies -> butterfly
fishing -> fish
signaling -> signal
```